Colección Electrónica

GUIA DEL BASIC

PARA PRINCIPIANTES



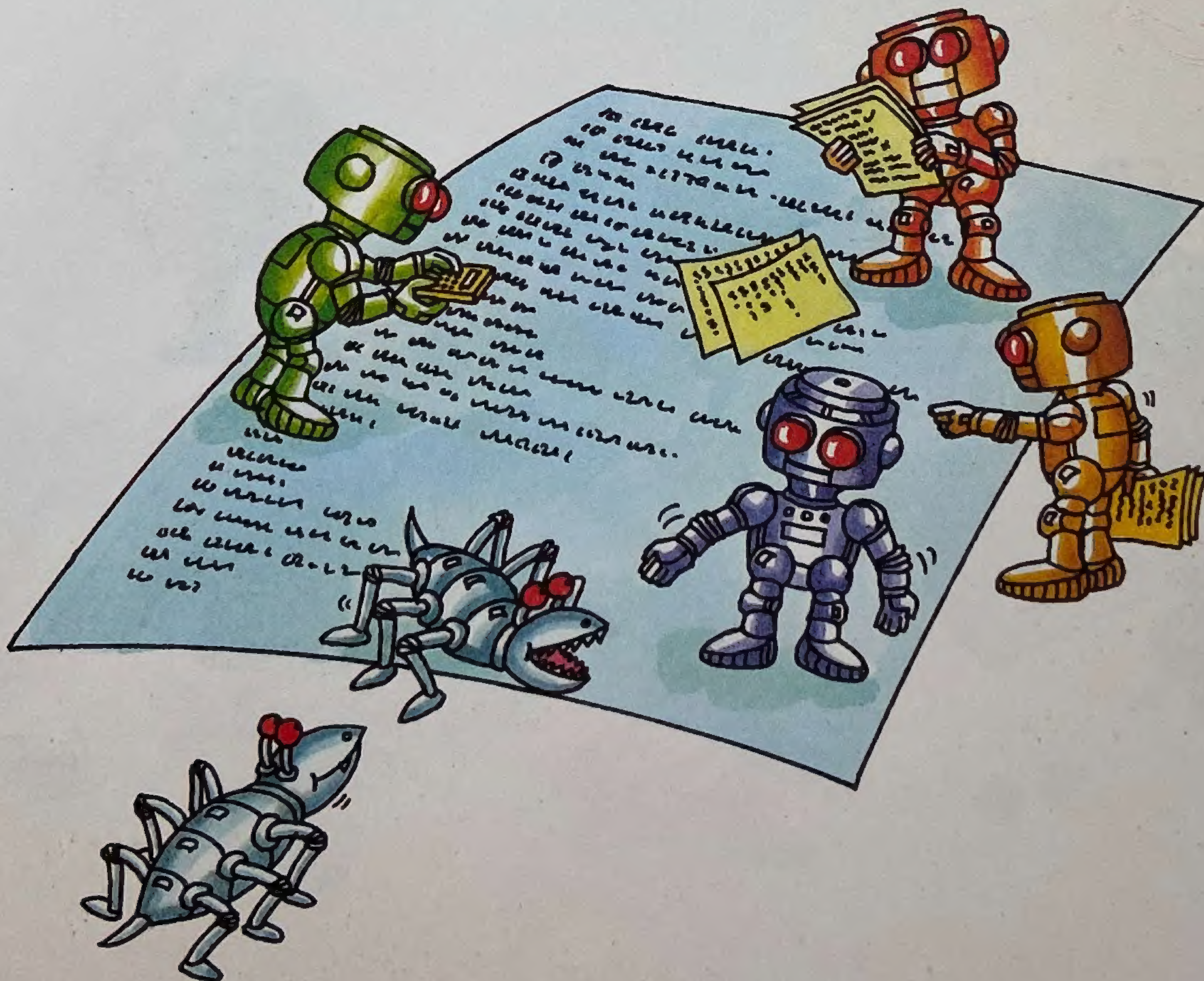
CON MUCHOS PROGRAMAS TIPO

Ediciones Plesa sm

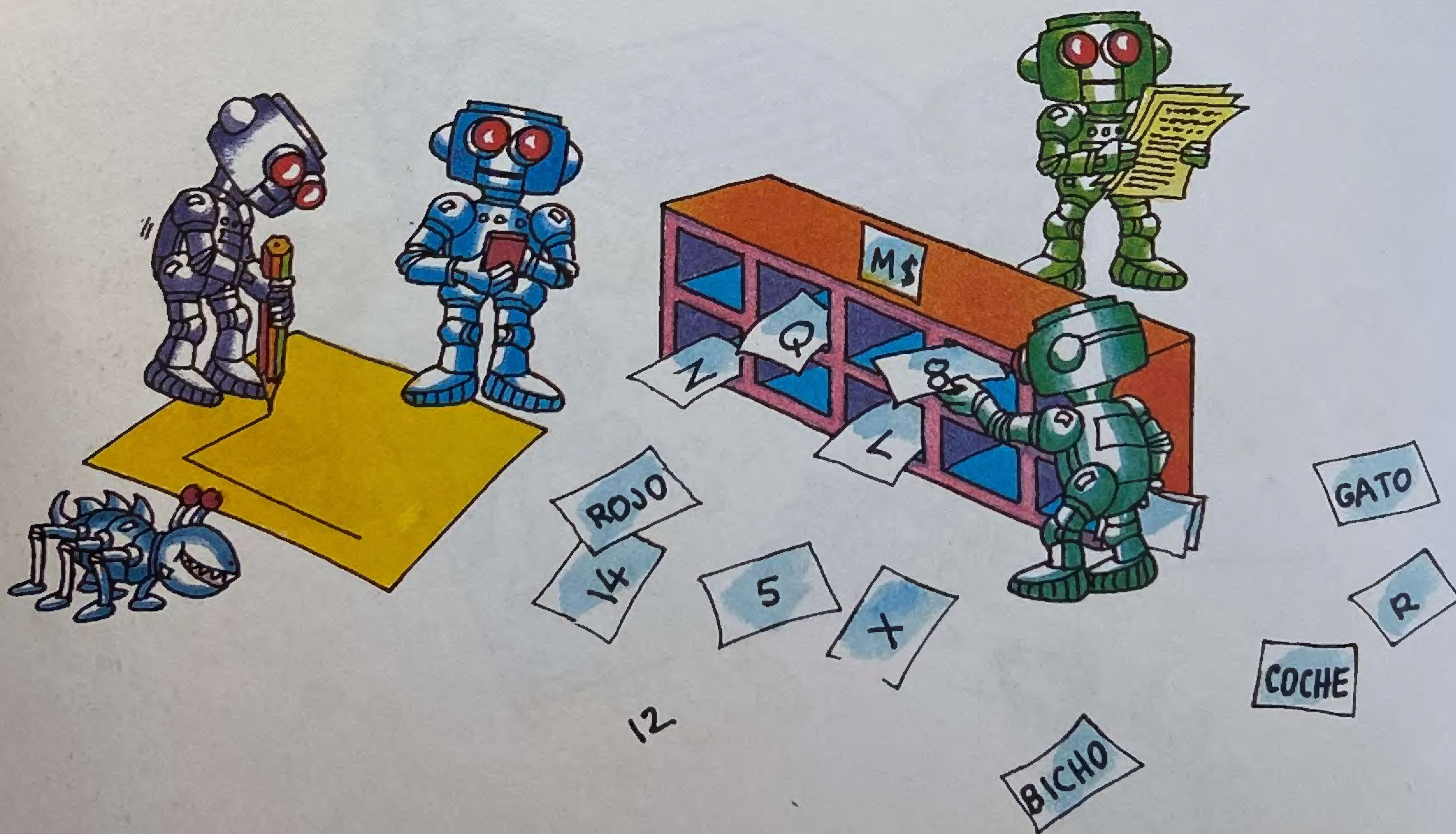
Colección Electrónica

GUIA DEL BASIC

Brian Reffin Smith
y Lisa Watts



Contenido



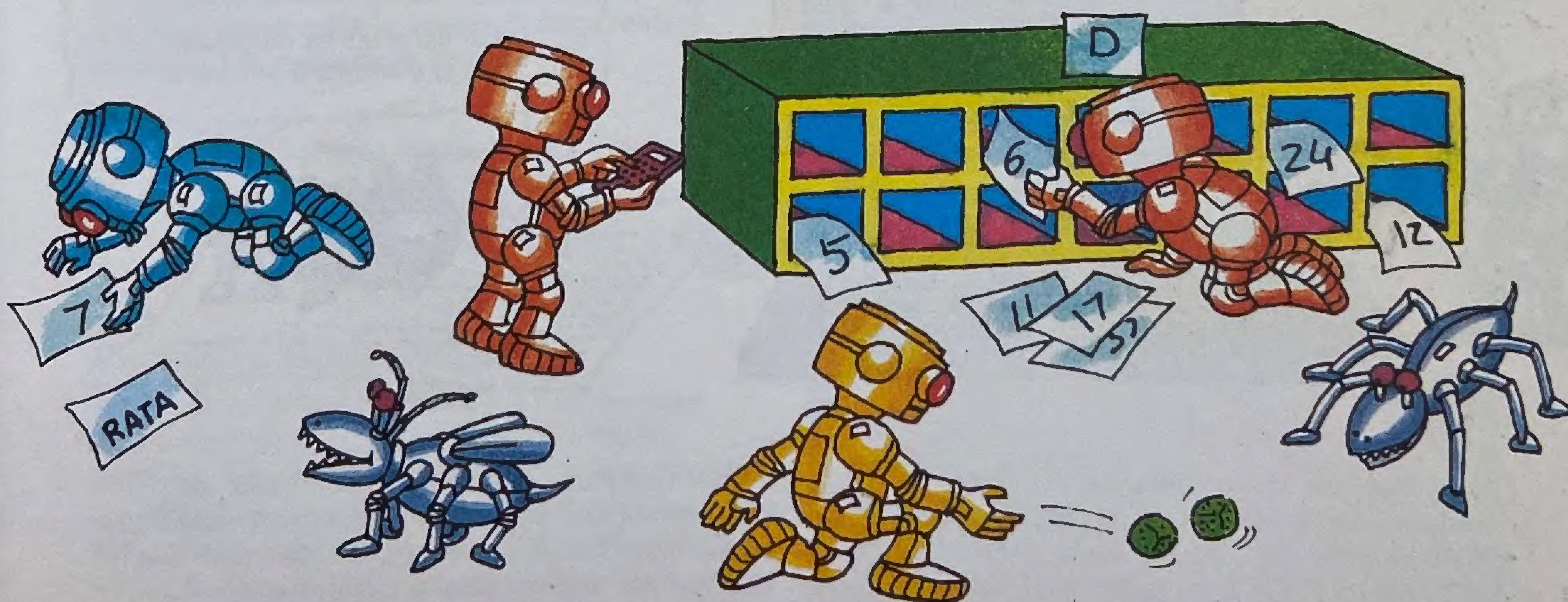
Sobre este libro

Este libro es una guía para comprender programas y mejorar tu BASIC paso a paso. No todo el mundo quiere escribir sus propios programas, pero una vez que entiendes el BASIC, es fácil adaptar o desparasitar programas de otras personas; y desde ahí hay un corto paso a que escribas los tuyos.

Al principio del libro hay una corta guía sobre los principales comandos en BASIC, con muchos ejemplos para que veas cómo funcionan. A continuación el libro enseña cómo se usan los comandos en programas para hacer cosas bastante complicadas, cómo crear un banco de datos, dibujar modelos en la pantalla y ordenar datos.

Los programas están escritos en BASIC «standard», esto es, una versión de BASIC que, con pequeños cambios, funcionará en la mayor parte de las computadoras. Hay una guía para transformar los programas, de modo que funcionen en tu computadora en las páginas 10-11 y las conversiones para las computadoras Sinclair que usan un BASIC ligeramente no standard, aparecen al final del libro.

A lo largo de todos los programas hay detalladas explicaciones de cómo funcionan y de técnicas y rutinas útiles que tú podrías usar en tus propios programas. También hay muchas ideas para experimentar con los programas y adaptarlas para que lleven a cabo diferentes trabajos.

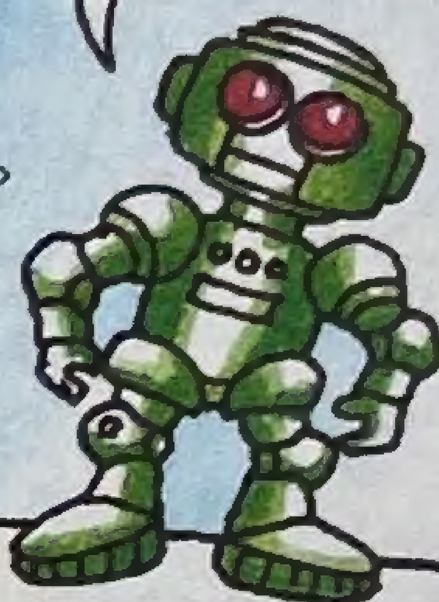


Introducción al BASIC

El lenguaje de programación BASIC consta de unas cien palabras. Cada palabra es una instrucción que le dice a la computadora que haga algo. Para hacer que la computadora realice un determinado trabajo tú le das una lista de instrucciones y la información sobre la que trabajar, es decir, un programa. Sólo puedes usar palabras del BASIC como instrucciones y también debes seguir las reglas, llamadas sintaxis, del lenguaje.

En BASIC cada línea de instrucciones tiene un número. Los números normalmente van de 10 en 10 para que puedas añadir otras líneas sin tener que renumerar todo el programa.

```
10 CLS
20 PRINT «DESPEGUE DE LA NAVE ESPACIAL»
30 LET G=INT (RND(1)*20+1)
40 LET W=INT (RND(1)*40+1)
50 LET R=G*W
60 PRINT «GRAVEDAD=»; G
70 PRINT «POR FAVOR, INTRODUCE LA FUERZA»
80 FOR C=1 TO 10
90 INPUT F
100 IF F>R THEN PRINT «MUY ALTO»;
110 IF F<R THEN PRINT
120 IF F=R THEN GOTO
130 IF C<>10 THEN PR
140 NEXT C
150 PRINT
160 PRINT «FALLASTE—»
170 PRINT «LOS INTRUSOS SE...»
180 STOP
190 PRINT «BUEN DESPEGUE»
```



Introducción de un programa

Cuando estás introduciendo un programa en la computadora tienes que teclear RETURN (o ENTER o NEWLINE, varía en las diferentes computadoras) al final de cada línea. Esto hace que la computadora guarde esa línea en su memoria y espere a la siguiente; cuando hayas introducido todas las líneas del programa teclea RUN. Esto le dice a la computadora que lleve a cabo las instrucciones.

PRINT es la instrucción que le dice a la computadora que escriba algo en la pantalla, en este caso la palabra PLATANOS.

PRINT «PLATANOS»
PLATANOS
>

Este es el cursor. Indica dónde aparecerá el próximo carácter.



Si tecleas una instrucción sin número entonces la computadora la llevará a cabo directamente, tan pronto como presiones RETURN (o ENTER o NEWLINE), esto se llama un comando directo. Por ejemplo, para decirle a la computadora que exponga las líneas del programa que le has metido, tecleas LIST como un comando directo. Para borrar la pantalla en la mayoría de las computadoras se teclea CLS.

Hay que tener cuidado para teclear los programas con precisión si escribes cualquier palabra de BASIC, o tecleas letras, números o puntuación equivocada la computadora no será capaz de seguir las instrucciones. Un error en un programa se llama parásito. La mayor parte de los parásitos son por errores al teclear, pero a veces son errores en la lógica de un programa y pueden conducir a resultados sorprendentes.

Guía de BASIC

En las próximas páginas hay una guía sobre los comandos BASIC más importantes y cómo usarlos. Si tú tienes una computadora, deberías comprobar los comandos en tu manual, ya que algunas palabras y reglas varían ligeramente en las diferentes computadoras.

PRINT ▶

Dile a la computadora que exponga algo en la pantalla, las letras y los símbolos deben ir entre comillas, pero con los números no es necesario, como se muestra en los ejemplos a la derecha. En estos ejemplos no hay número de línea, así que la computadora lleva a cabo cada instrucción tan pronto presiones RETURN. Escribirá exactamente lo que hayas tecleado entre las comillas, incluyendo espacios. La palabra PRINT sola en una línea le dice a la computadora que deje una línea en blanco.

```
PRINT 254
254
PRINT 999
999
PRINT

PRINT «***CEBRAS***»
***CEBRAS***
PRINT «RATAS»
RATAS
```

La palabra PRINT por sí misma fabrica esta línea vacía.

Haciendo cálculos ▶

También puedes usar PRINT para obtener las respuestas a cálculos. La computadora usa los signos usuales para sumar y restar, pero para multiplicar usa un * y para dividir un signo /. SQR(N) es la instrucción que encuentra la raíz cuadrada de un número, N y ↑, o ∧, o ** significa elevado en la potencia de. Por ejemplo, 3 ↑ 2 significa 3 elevado a la potencia de 2, ó 3 al cuadrado.

En operaciones complicadas la computadora siempre realiza las multiplicaciones y divisiones antes de sumar o restar. Para evitar esto puedes usar paréntesis para indicar a la computadora en qué orden tiene que realizar las sumas. En cálculos con muchos paréntesis la computadora realiza primero los paréntesis interiores.

```
PRINT 12095+277
12372
PRINT 239-51
188
PRINT 17*5
85
PRINT 221/13
17
```

```
PRINT SQR(9)
3
PRINT SQR(9)+3 2
12
PRINT 2*17-5
29
PRINT 2*(17-5)
24
```



Comas, y punto y coma▶

Estos dicen a la computadora dónde tiene que imprimir el próximo carácter en la pantalla. Un punto y coma le dice que no deje ningún espacio y la coma le dice que se mueva adelante un cierto número (el número varía en las diferentes computadoras).

Algunas computadoras necesitan una coma o punto y coma para separar las sentencias PRINT y los datos o variables (letras que representan datos en la memoria de la computadora). Prueba con los ejemplos de la derecha para ver cómo funciona en tu computadora.

```
PRINT «JUNTOS»; «Y»; «SEPARADOS»
JUNTOS    SEPARADOS
PRINT «TOTAL=»; 2*17
TOTAL=34
PRINT «TOTAL=», 2*17
TOTAL=          34
PRINT 12, 24
12          24
```

La coma hizo que la computadora dejara este espacio.

Variables ►

La información que le das a la computadora para que trabaje sobre ella se llama datos (data). Cuando le des a la computadora algún dato para que lo guarde en su memoria también tienes que darle un nombre. El nombre se llama variable, y cuando quieres que la computadora haga algo con el dato que tú le has dado te refieres a él por su nombre de variable, se llama variable porque el dato al que se refiere puede cambiar durante el programa.

Se usan letras del alfabeto, o una letra y un número, por ejemplo A6, como nombres para datos numéricos, un dato consistente en letras y símbolos se llama cadena, y para cadena se usan letras del alfabeto o una letra y un número seguidos del signo dólar, por ejemplo P$ (pronunciado P dólar o P string) o P6$. Las diferentes computadoras tienen reglas distintas para los nombres de variables, así que compruébalo en el manual de tu computadora.



LET ►

Es una forma de introducir datos en la computadora. LET A5 dice a la computadora que guarde el número 5 en su memoria y que lo llame A y LET C$=«CONEJOS» guarde la cadena de letras en un espacio de la memoria bajo el nombre de C$. Las cadenas siempre tienen que estar entre comillas pero los números no lo necesitan.

```
10 LET A=5
20 LET C$=«CONEJOS»
30 PRINT A
40 PRINT C$
RUN
5
CONEJOS
```



Puedes usar PRINT con el nombre de una variable para decirle a la computadora que exponga el dato en la pantalla.

INPUT ►

Este es un modo de introducir datos en la computadora mientras está llevando a cabo un programa. La palabra INPUT va seguida por el nombre de una variable. Cuando la computadora llega a un comando INPUT imprime un signo de interrogación (u otro símbolo) en la pantalla y espera a que teclees el dato. Si la variable de INPUT es una variable numérica deberás darle un dato numérico y si es una variable de cadena debes introducirle una cadena.

En la mayoría de las computadoras puedes poner palabras entre comillas en una instrucción INPUT, para hacerlo más claro, como se muestra en el segundo ejemplo a la derecha. No uses este método en la computadora VIC, ya que el VIC almacenará las palabras en la variable junto con el dato que introduzcas. La mayoría de las computadoras necesitan un punto y coma entre las palabras y el nombre de la variable.

```
10 PRINT «¿COMO TE LLAMAS?»
20 INPUT N$
RUN
¿COMO TE LLAMAS?
?JACK
```

Signo de interrogación de la computadora.

La respuesta de la persona almacenada en N$.

```
10 INPUT «¿COMO TE LLAMAS?»; N$
20 INPUT «¿CUANTOS AÑOS TIENES?»; A
RUN
¿COMO TE LLAMAS? JACK
¿CUANTOS AÑOS TIENES? 21
```



Signos de interrogación de la computadora.



La mayoría de las computadoras tienen teclas especiales para borrar los caracteres tecleados por equivocación. Para corregir una línea en un programa puedes escribir la línea entera otra vez incluyendo el número de línea. La nueva línea sustituirá a la línea con el error. Para borrar una línea completamente teclea el número de línea solamente.



READ/DATA ▶

Es otro método de darle información a la computadora, la palabra READ está seguida por uno o más nombres de variables mientras la información de las variables está en una línea que empieza con la palabra DATA. La información puede estar en cualquier lugar del programa. Cuando la computadora encuentra la instrucción READ busca la palabra DATA e introduce cada uno de los elementos en orden dentro de las variables los elementos de información han de ir separados por comas y en algunas computadoras las cadenas de información han de ir entre comillas. Otras sólo necesitan comillas si la cadena contiene espacios o puntuación.

```
10 READ A$, B, C$, N
.
.
.
100 DATA RUEDAS, 2000
110 DATA «CONDUCIR COCHES», 500
```

Este elemento de información está entre comillas porque contiene espacios.

IF/THEN ▶

Esta es una manera de comprobar información, y de decirle a la computadora que haga ciertas cosas de acuerdo con el resultado. Puedes comprobar si dos datos son iguales, distintos o si uno es mayor o menor que el otro usando los signos expuestos a la derecha. Prácticamente todas las instrucciones pueden ir detras de la palabra THEN, pero si la comprobación no es verdadera ignora el THEN y continua con el resto del programa.

```
IF A=B THEN PRINT «IGUAL»
IF X<>Y THEN PRINT «DISTINTO»
IF X>Y THEN PRINT «X MAYOR»
IF X<Y THEN PRINT «X MENOR»

IF A$=«NO» THEN STOP
IF X+Y=5 THEN LET X=X+1
```

Esto le dice a la computadora que deje de ejecutar el programa.

GOTO ▶

Le dice a la computadora que vaya a otra línea del programa. Se usa normalmente con IF/THEN para que la computadora solo se bifurque si se dan ciertas condiciones. Ten cuidado cuando uses GOTO por sí solo, ya que puede crear un bucle continuo como se muestra en la línea 185 a la derecha; la única manera de detener el programa sería teclear BREAK o ESCAPE (varía en las diferentes computadoras).

```
100 IF A=5 THEN GOTO 175
.
.
.
175 PRINT «CONGRATULACIONES»
180 PRINT «****HAS GANADO****»
185 GOTO 180
```

Evitar usar GOTO para hacer un bucle continuo como en la línea 185.

GOSUB/RETURN ▶

GOSUB hace que la computadora vaya a una subrutina, una parte especial del programa para llevar a cabo una tarea específica. La palabra RETURN al final de la subrutina manda a la computadora de nuevo a la instrucción justo después del comando GOSUB. Obtendrás un error si te olvidas de la palabra RETURN.

REM ▶

Esta es la abreviatura de «reminder» (recordatorio) o «remark» (nota). La computadora ignora aquellas líneas que empiecen por la palabra REM y es útil para introducir notas en el programa que te recuerden que está ocurriendo.

```
100 INPUT «¿QUIERES JUGAR?»; T$
110 IF T$=«SI» THEN GOSUB 500
120 REM COMIENZA EL JUEGO
.
.
500 REM HOLA, SUBRUTINA
510 PRINT «¿COMO TE LLAMAS?»
520 INPUT N$
530 PRINT «HOLA»; N$
540 RETURN
```

La computadora ignora esta línea.

La línea 540 manda a la computadora a la instrucción inmediatamente después de GOSUB.

Bucles FOR/NEXT ▶

Las palabras FOR, TO y NEXT hacen que la computadora repita una parte del programa un cierto número de veces. En el ejemplo de la derecha, las líneas 10 a 30 se repiten 3 veces y cada vez la computadora imprime el texto de la línea 20. J es una variable para contar el número de repeticiones. La línea 30 manda a la computadora a buscar el próximo valor de J y cada vez que el bucle se repite se añade uno a J. Cuando J = 3 la computadora continua con el resto del programa.

```
10 FOR J=1 TO 3
20 PRINT «J LOOP»; J
30 NEXT J
40 PRINT
RUN
J LOOP 1
J LOOP 2
J LOOP 3
```

La variable J al final de la línea hace que la computadora imprima el valor de J cada vez que el bucle se repite.

STEP ▶

Esto cambia la manera en la que J cuenta el número de bucles. Por ejemplo FOR J = 1 TO STEP 2 hace que J se incremente en 2 cada vez y STEP X le haría incrementarse en el valor que hubiera almacenado en X. En el ejemplo de la derecha STEP − 1 hace que J cuente hacia atrás.

```
10 FOR J=10 TO 1 STEP−1
20 PRINT J; «DIAS PARA NAVIDADES»
30 NEXT J
40 PRINT «FELICES NAVIDADES»
RUN
10 DIAS PARA NAVIDADES
9 DIAS PARA NAVIDADES
8 DIAS PARA NAVIDADES
7 DIAS PARA NAVIDADES
```

Bucles de anidamiento ▶

Puedes hacer repeticiones bastante complicadas usando bucles dentro de bucles. Estos se llaman bucles de anidamiento. Por ejemplo, en el programa de la derecha cada vez que el bucle desde las líneas 10 a 50 es repetido el bucle de las líneas 20 a 40 se repite 12 veces. Cada vez que se repite el bucle interno de la línea 30 imprime el valor de J × I.

Ambas partes del bucle interno han de estar dentro del bucle externo u obtendrás un error.

```
10 FOR I=2 TO 12
20 FOR J=1 TO 12
30 PRINT J; «VECES»; I; «=»; J*I
40 NEXT J
50 NEXT I
```

Bucle externo.

Comandos de gráficos ▶

La computadora hace dibujos iluminando pequeños cuadrados llamados pixels, en la pantalla. La instrucción para iluminar pixels varía con las diferentes computadoras. Los programas en este libro usan la instrucción PLOT X, Y donde X e Y son las coordenadas del pixel. Para dibujar una línea los programas usan DRAW X, Y. La mayoría de las computadoras tienen instrucciones similares, pero algunas pueden necesitar una instrucción adicional para que les digan que modalidad de gráficas quieres*.

El número de pixels que la computadora puede iluminar a lo largo de la pantalla se llama ancho de la pantalla y el número hacia arriba o hacia abajo en algunas computadoras es la altura de la pantalla.

X

Y

RND ▶

Hace que la computadora genere un número aleatorio aunque la instrucción precisa varía en diferentes computadoras. En algunas RND(9) genera un número entre 1 y 9. Otras necesitan una instrucción más complicada como esta: INT(RND(1) * 9 + 1) la computadora calcula todo lo que está dentro del paréntesis primero. RND(1) hace que genere un número entre 0 y 1. Multiplica este por 9, el mayor número que quiere, entonces le añade 1 ya que la palabra INT le hace un número entero por defecto.

```
PRINT RND(9)
7
PRINT INT (RND(1)*9+1)
7
```

Algunas computadoras usan RND(0) en vez de RND(1).

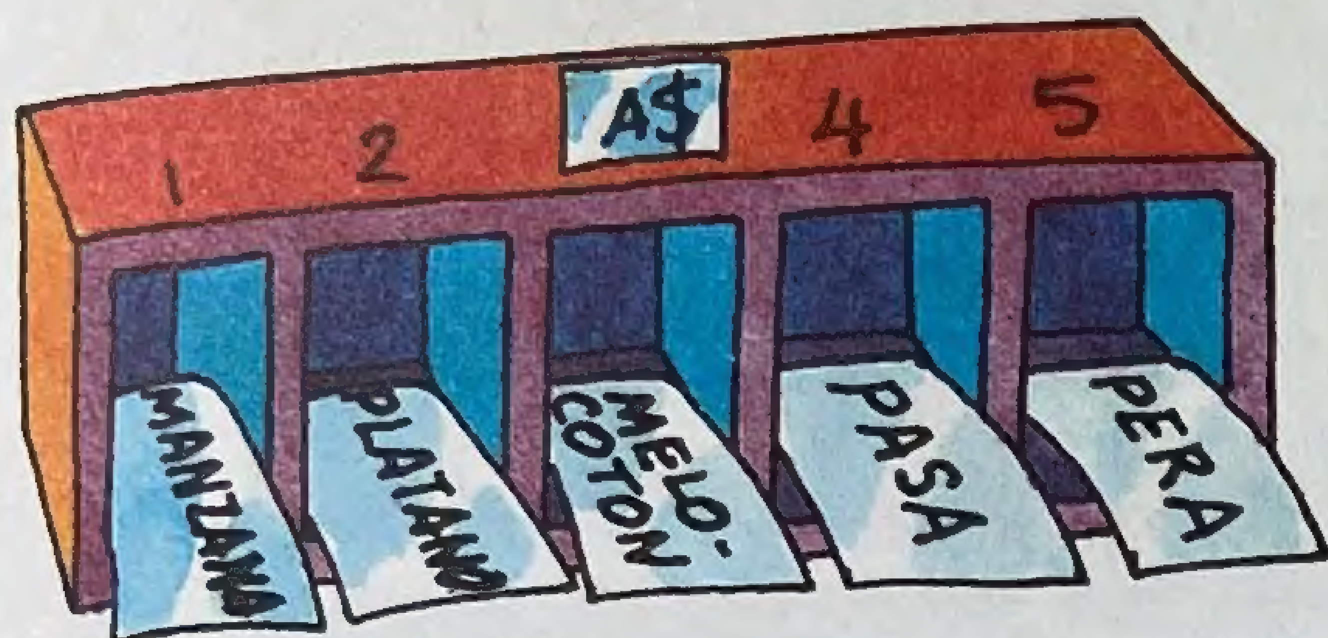
0,0109425*9 + 1 =

* La mayoría de las computadoras tienen varios «modos» y en cada modo pueden trabajar con un número diferente de colores y pixels.

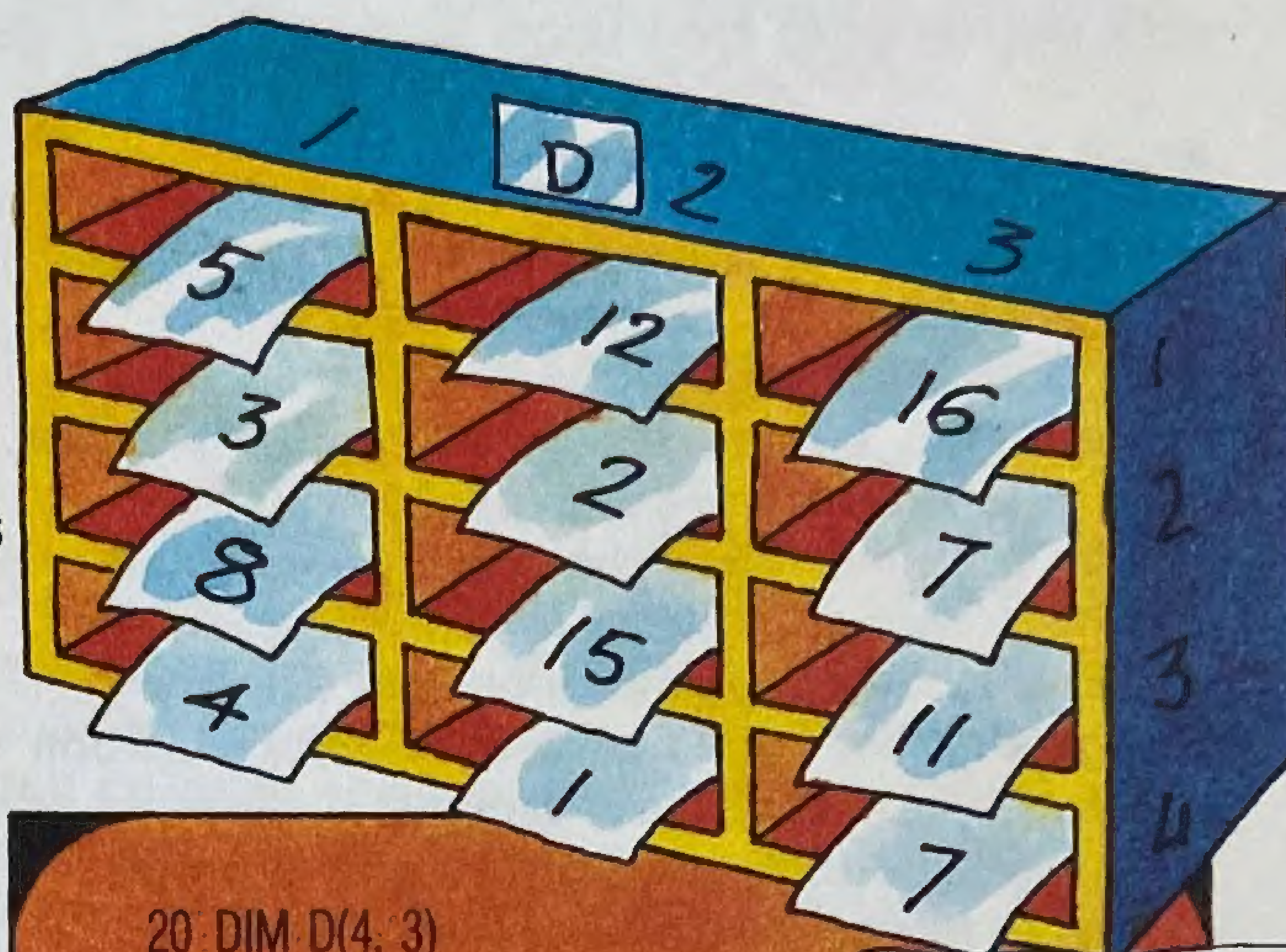
Matrices ▼

Una matriz es un conjunto de elementos de información almacenados juntos bajo el nombre de una variable. Puedes imaginarte la variable como un espacio en la memoria de la computadora con un gran número de compartimentos. Las matrices pueden ser unidireccionales, esto es, una sóla fila de cajas, o bidimensionales y tener varias filas de cajas. Tu localizas un elemento en una matriz unidimensional por el número de la caja en que está, ej.: en el dibujo de abajo A$(4) es PASA. Para matrices bidimensionales tienes que dar el número de la fila y la columna, ej.: D(3, 2) es 15. Los números entre paréntesis se llaman subíndices.

DIM A$(5) y DIM D(4,3)







Antes de usar una matriz tienes que decirle a la computadora lo grande que es usando la palabra DIM (abreviatura de dimensión) como se muestra a la derecha. Para introducir datos en una matriz usas READ/DATA con un bucle. Para matrices bidimensionales necesitas bucles de anidamiento como se muestra a la derecha.

En este ejemplo I es el número de fila y J es el número de columna. Cada vez que se repite el bucle interior J pone información en la siguiente columna de la fila. Cuando se repite el bucle I la computadora empieza con una nueva fila.

```
20 DIM D(4, 3)
30 FOR I=1 TO 4
40 FOR J=1 TO 3
50 READ D(I, J)
60 NEXT J
70 NEXT I
80 DATA 5, 12, 16
90 DATA 3, 2, 7
100 DATA 8, 15, 11
110 DATA 4, 1, 7
```

La sentencia DIM debería estar al principio del programa ya que sólo ha de ser usada una vez.



LEFT$ y RIGHT$ ►

Se utilizan para hacer cosas con los caracteres dentro de las variables de cadena. Por ejemplo, LEFT$(A$, 4) le dice a la computadora que tome 4 caracteres de la izquierda de A$ y RIGHT$(A$, 5) significa que tome 5 de la derecha. Las computadoras Sinclair no usan estos comandos. Para ver las instrucciones usadas en computadoras Sinclair ver página 11.

```
10 LET A$=«PARA KEET»
20 PRINT LEFT$(A$, 4)
25 PRINT LEFT$(A$, 2)
30 PRINT
40 PRINT RIGHT$(A$, 5)
RUN
PARA
PA

AKEET
```

MID$ y LEN ►

MID$ le dice a la computadora que coja caracteres del centro de una cadena y LEN te dice a ti cuántos caracteres incluyendo espacios y puntuación hay en una cadena. Por ejemplo, MID$(K$,2,4) significa tome 4 caracteres del centro de K$ empezando por el segundo carácter. Ver en página 11 las instrucciones para usarlos en computadoras Sinclair.

```
10 LET K$=«DISFRAZAR»
20 PRINT MID$(K$, 2, 4)
25 PRINT MID$(K$, 4, 4)
30 PRINT
35 PRINT LEN(K$)
RUN
RAVE
VEST

8
```



Este es el número de caracteres en K$.

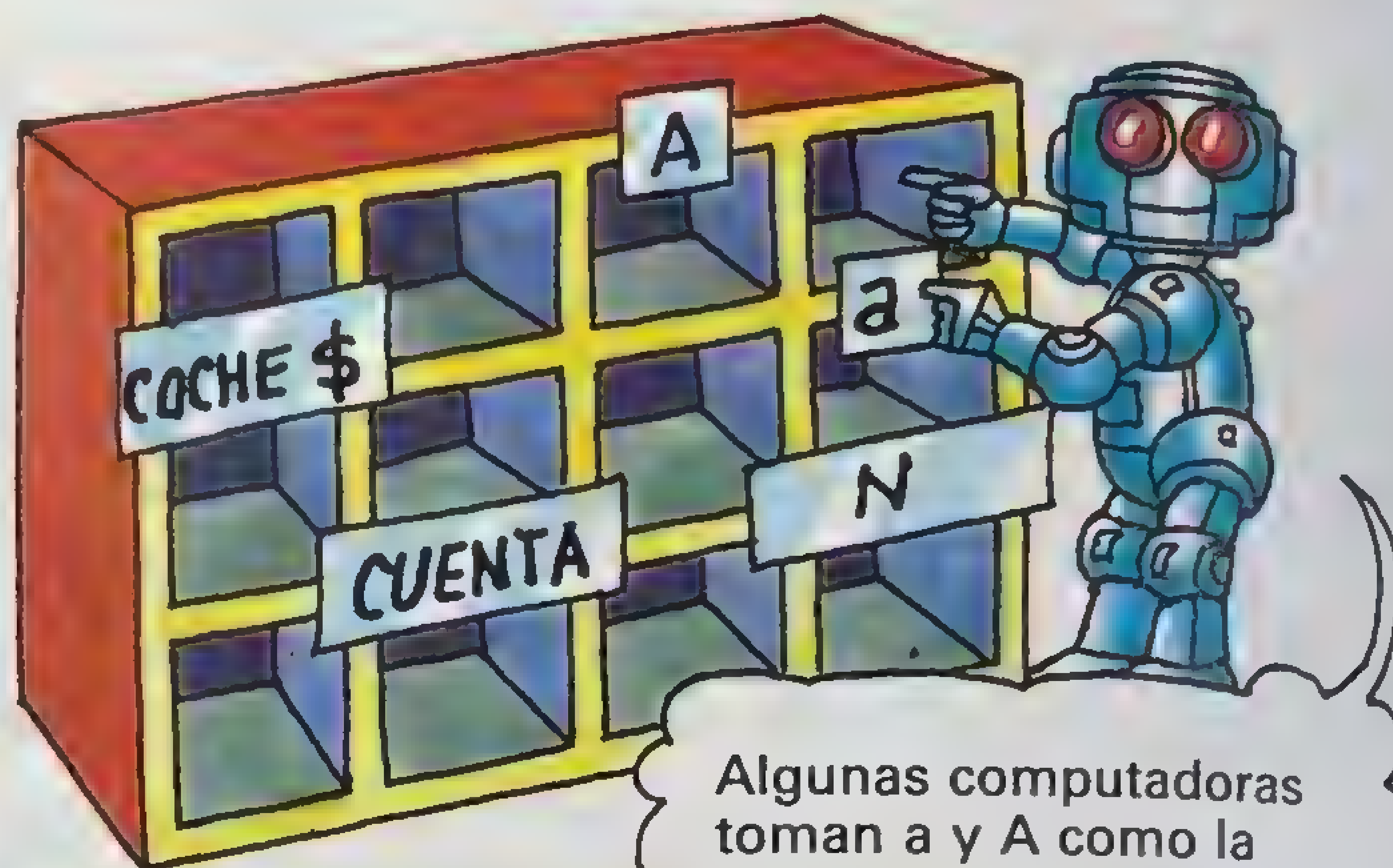
El BASIC en este libro

Los programas en este libro están escritos en BASIC standard. Algunas computadoras, no obstante, tienen sus propios sistemas especiales para hacer cosas por lo que tendrás que hacer algunos pequeños cambios para hacer funcionar los programas en tu computadora. En estas dos páginas hay algunos puntos que deberías tener en cuenta.

Los programas están escritos para funcionar en muchas marcas de computadora diferentes por lo que no tienen en cuenta las características especiales de alguna máquina en particular. Una vez que sepas como funcionan los programas ya podrás adaptarlos para que usen algunas de las características especiales de tu computadora.

Nombres de variables ▶

Algunas computadoras pueden usar palabras como nombres de variables mientras que otras solo aceptarán letras o letras y dígitos. Por ejemplo, en computadoras Sinclair puedes usar palabras cortas para nombres de variables numéricas pero sólo se te permite usar una letra para nombres de variables de cadena. En los programas en este libro la mayoría de las variables están nombradas con palabras para hacer más fácil su comprensión. Si tu computadora no acepta palabras usa sólo la primera letra de la palabra para el nombre de la variable.



Algunas computadoras toman a y A como la misma variable.

LET ▶

La mayoría de las computadoras no necesitan la palabra LET en la sentencia LET FRUTA$ = «MANZANA». Otras tampoco necesitan THEN en sentencias IF ... THEN. Todos los programas en este libro usan LET y THEN pero los puedes excluir si tu computadora no los necesita.

```
100 FRUTA$=«MANZANA»
110 CUENTA=CUENTA+1
120 IF CUENTA=10 PRINT «PREPARADO»
```

Iniciando variables ▶

En algunas computadoras tienes que fijar o inicializar una variable antes de que puedas usarla. Esto significa que tienes que darle un valor a la variable al principio del programa como se muestra a la derecha. Otras supondrán que una variable numérica es 0 y una variable de cadena está vacía si no están inicializadas. Los programas en este libro incluyen líneas para inicializar variables pero los puedes dejar fuera si tu computadora no los necesita.

```
10 LET A=0
20 LET FRASE$=»»
.
.
.
.
100 LET FRASE$=FRASE$+«K»
110 LET A=A+1
```



Una cadena vacía se llama cadena nula.

INPUT ▶

La mayoría de las computadoras aceptarán palabras entre comillas con sentencias INPUT. La diferencia puede estar en que necesiten un punto y coma delante de la variable a introducir o si automáticamente dejan un espacio entre las palabras y los datos que introduces. Puedes averiguar lo que necesita tu computadora experimentando o mirando en tu manual.

```
10 INPUT «¿COMO TE LLAMAS?»; N$
20 PRINT «HOLA»; N$
```



En algunas computadoras necesitas un espacio dentro de las comillas o los datos saldrán pegados a las palabras.

* No uses este método en la computadora VIC ya que pondrá en la variable tanto las palabras como los datos.

DATA

```
100 DATA RATON, JERBO, RATA
110 DATA GNU, «PEREZOSO DE TRES DEDOS»
120 DATA «CIERVO, ROJO», «RINOCERONTE, NEGRO», GIRAFA
```



Elementos de información entre comillas que incluyen espacios y elementos de puntuación.

Ser especialmente cuidadoso al escribir líneas de datos. Cada elemento de información debe estar separado por una coma y es muy fácil cometer errores. Algunas computadoras también necesitan que las palabras estén entre comillas. Otras sólo necesitan comillas si la información incluye espacios o puntuación.

Líneas multisentenciales

```
500 PLOT 40, 1: DRAW 1, 1
180 IF A=10 THEN PRINT «CORRECTO»: GOTO 100
```

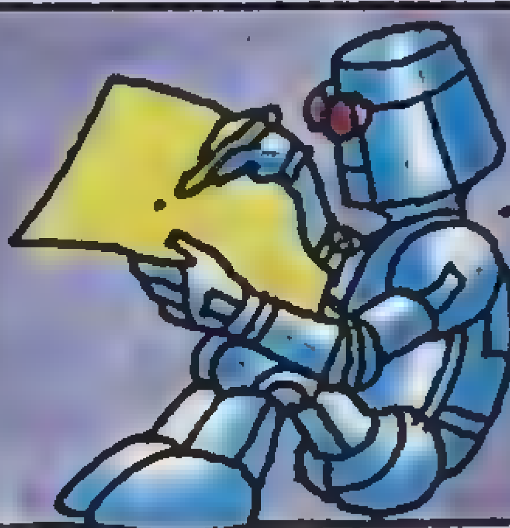


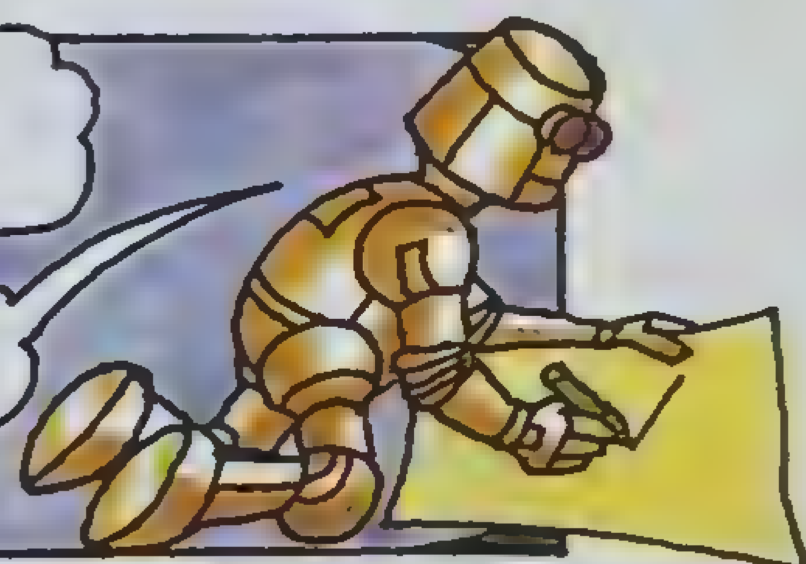
Esto sólo ocurre si A = 10.

La mayoría de las computadoras aceptan varias instrucciones en la misma línea separadas por dos puntos como se muestra abajo. Esto usa menos espacio en la memoria y puede hacer la lectura del programa más sencilla. Si tu computadora no acepta líneas multisentenciales pon cada instrucción en una línea diferente. Si estás usando líneas multisentenciales en tus propios programas ten cuidado al poner sentencias adicionales después de IF... THEN, ya que sólo serán llevadas a cabo si la condición IF es verdadera.

RND y comandos de gráficos







Estos comandos varían en todas las computadoras. En los programas de este libro la instrucción para producir un número aleatorio entre 1 y N (donde N es cualquier número) es INT(RND(1)*N + 1). Los comandos de gráficas son PLOT X, Y para un punto y DRAW X, Y para una línea. Tendrás que sustituir los comandos de tu computadora por todos estos si tu computadora necesita una instrucción especial para gráficos.

Computadoras Sinclair y Cadenas

Las computadoras Sinclair trabajan con cadenas de una manera no stándard. No usan LEFT$, RIGHT$ o MID$. En su lugar tienes que decirles cuáles son los caracteres que deseas tomar de una cadena. Por ejemplo, en la computadora Sinclair PRINT A$ (1 TO 4) es lo mismo que PRINT (LEFT$(A$,4) y PRINT A$(4 TO 8) es lo mismo que MID$(A$,4,5).

En matrices de cadenas cada cadena debe tener el mismo número de caracteres (puedes rellenar cadenas cortas con espacios) y cada carácter de una cadena es almacenado en un compartimento separado en la cadena. Algunos de los programas en este libro necesitan ser reformados para funcionar en computadoras Sinclair y las conversiones vienen dadas en las págs 46-47.

Cambios en los programas

Una vez que tengas un programa en funcionamiento y tengas una idea de cómo funciona, puedes cambiarlo y adaptarlo introduciendo datos diferentes o añadiendo color y sonido.

Cuando estés cambiando un programa comprueba cada línea cuidadosamente. Comprueba que tienes bucles suficientes pero no demasiados para leer los nuevos datos y recuerda cambiar también las sentencias DIM.

Aprender BASIC estudiando los programas

Una buena manera de aprender BASIC es estudiando los programas de otras personas y viendo cómo funcionan. Estudiando los programas de este libro puedes aprender a usar bucles y cadenas, a escribir programas de gráficos simples y las diferentes formas de almacenar y ordenar información. A primera vista algunos de los programas parecen realmente complicados. Si bien un programa complicado no es más que una larga lista de comandos en BASIC unidos con un orden. En estas dos páginas hay algunos consejos que te ayudarán a estudiar y comprender programas.

Estudio de un programa

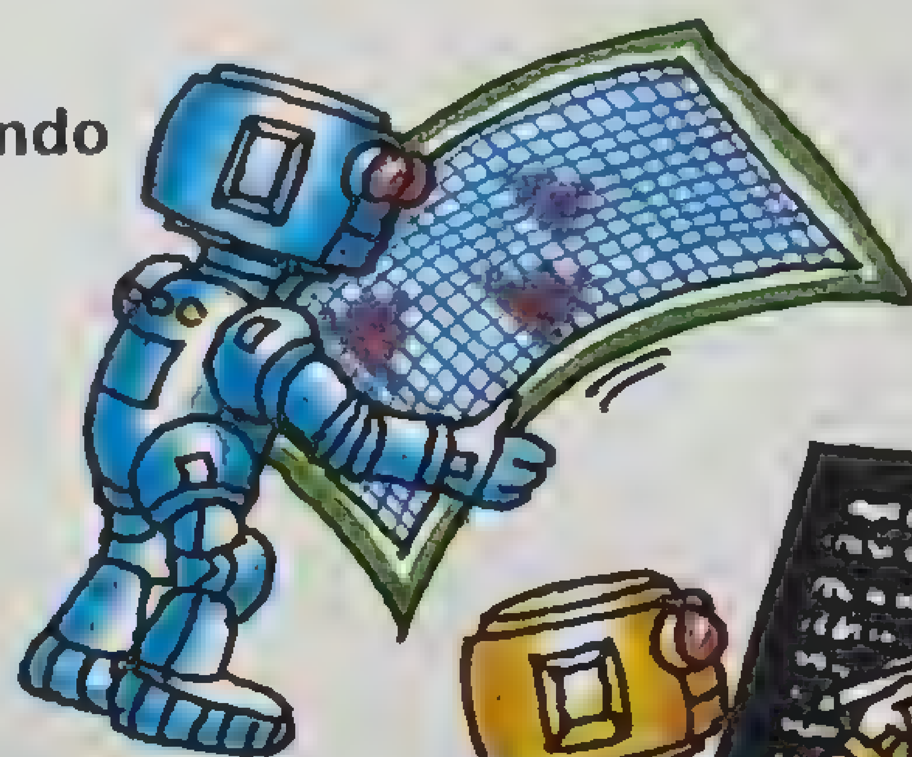
La mayoría de los programas están compuestos de varias partes diferentes (a veces llamadas rutinas o módulos) que llevan a cabo tareas diferentes. Por ejemplo, en un juego de persecución de cohetes una parte del programa será para dibujar los cohetes en la pantalla, el resto de las partes registrarán los ataques y blancos e indicarán el nivel de combustible, la velocidad e imprimirán las puntuaciones finales.

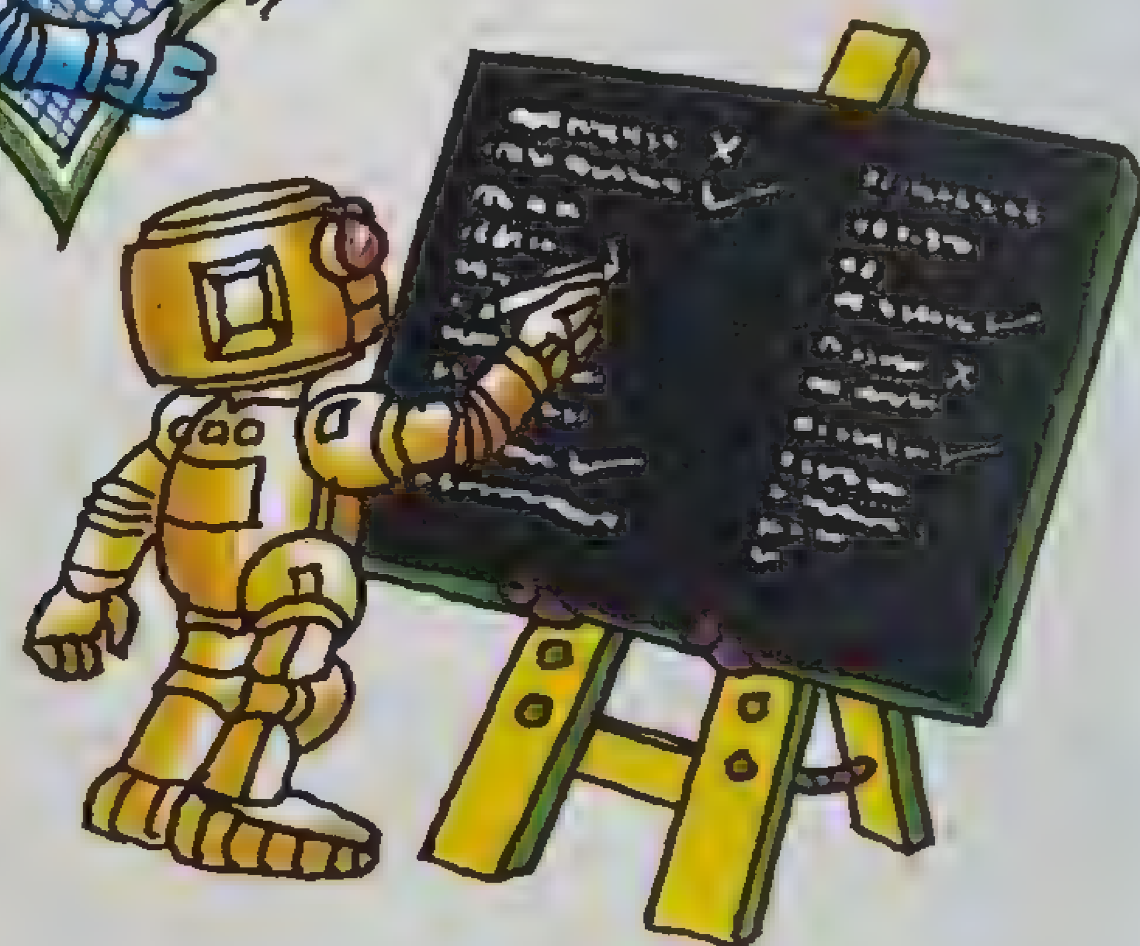
1. Dibujando cohetes.



3. Calculando los niveles de combustible y la velocidad.

2. Registrando ataques y blancos.

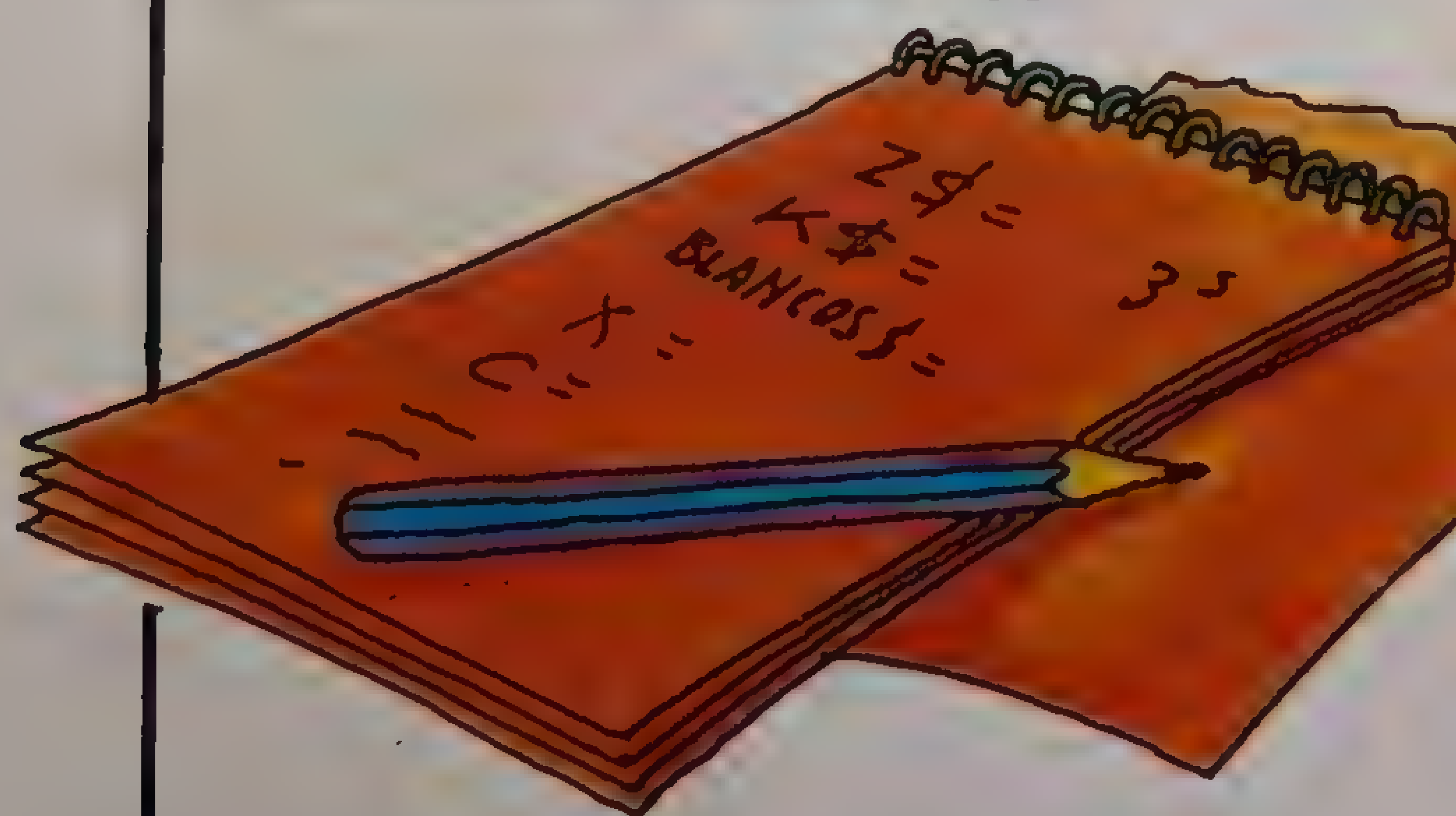


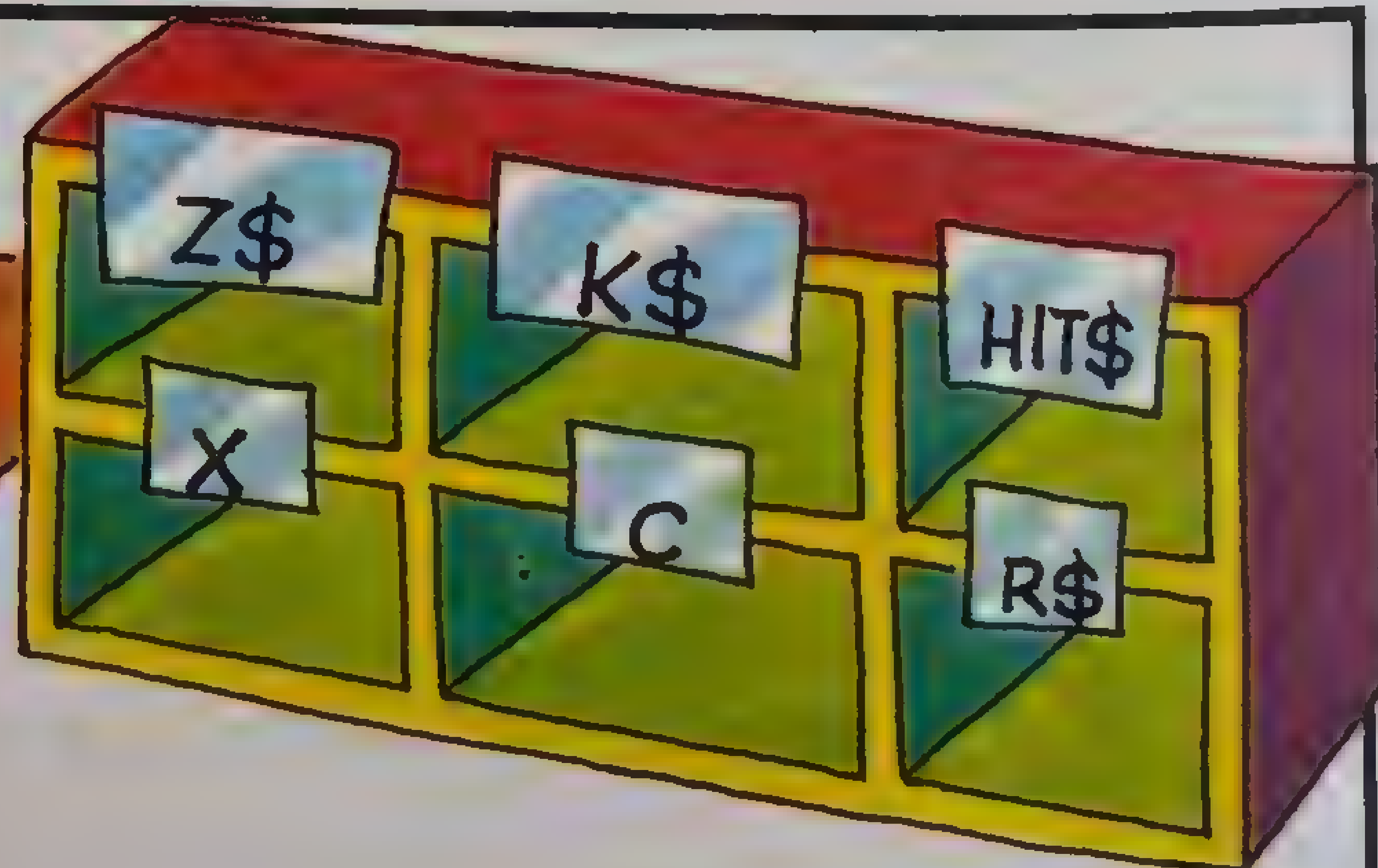


4. Escribiendo las puntuaciones.

El primer paso a la hora de estudiar un programa es intentar reconocer las diferentes partes y averiguar para qué sirven. Esto te da una idea general de cómo funciona un programa. Busca subrutinas que lleven a cabo trabajos determinados así como grandes saltos en los números de línea. Líneas que empiezan cien o mil números más adelante suelen indicar una nueva parte del programa. Algunas veces las diferentes partes del programa están indicadas con sentencias REM.

Descifrando variables

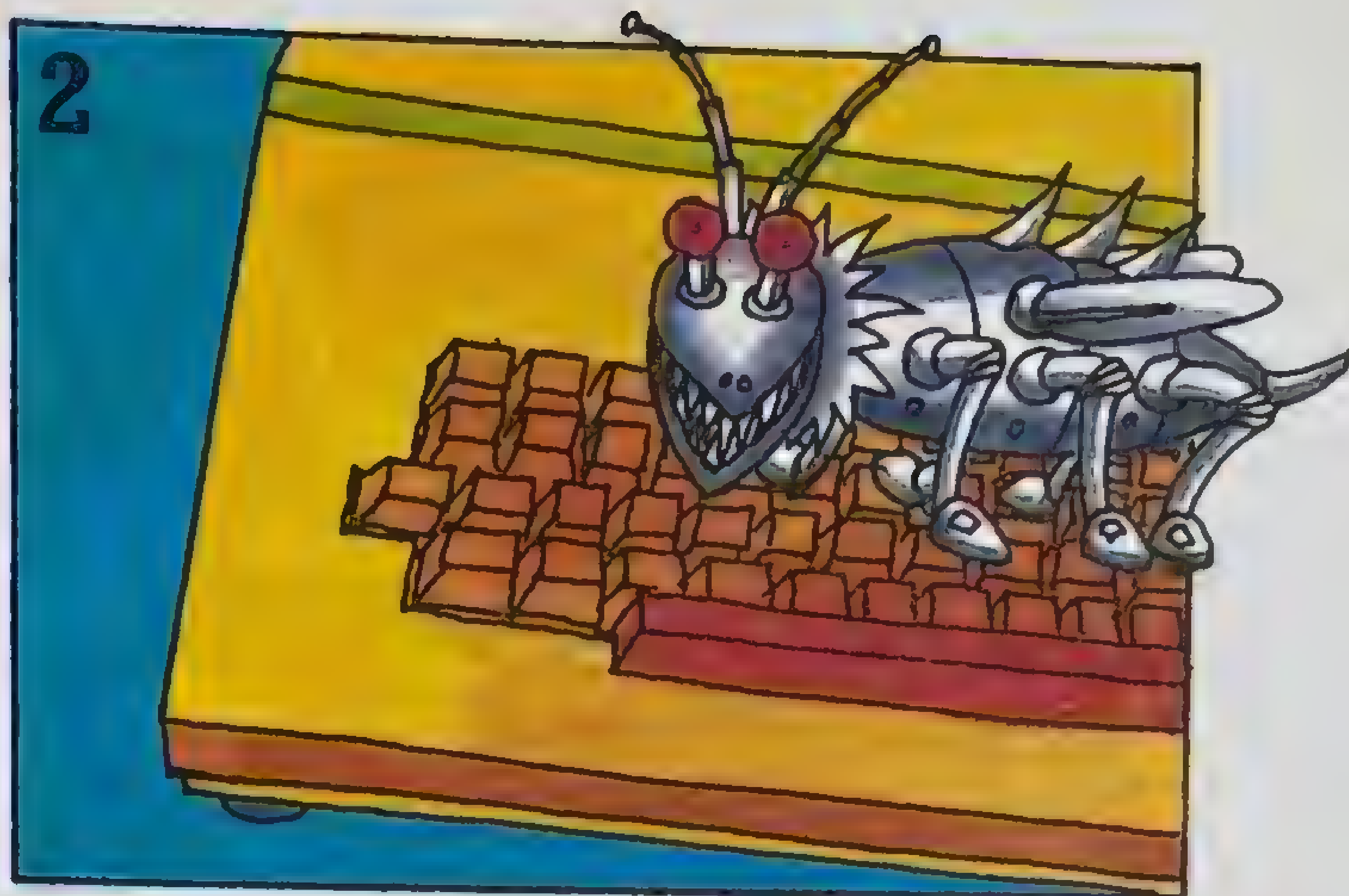




Probablemente la cosa más difícil de comprender en los programas sean las variables. Antes de que escribas un programa en una computadora es una buena idea decidir cuál será la función de cada variable y hacer nota de ellas. Algunas variables se usan normalmente para la misma tarea así que puedes reconocerlas inmediatamente. Por ejemplo, las letras I, J, K, L, se usan normalmente para bucles y Z o Z$ se usa para información que sólo será necesaria durante un corto período.



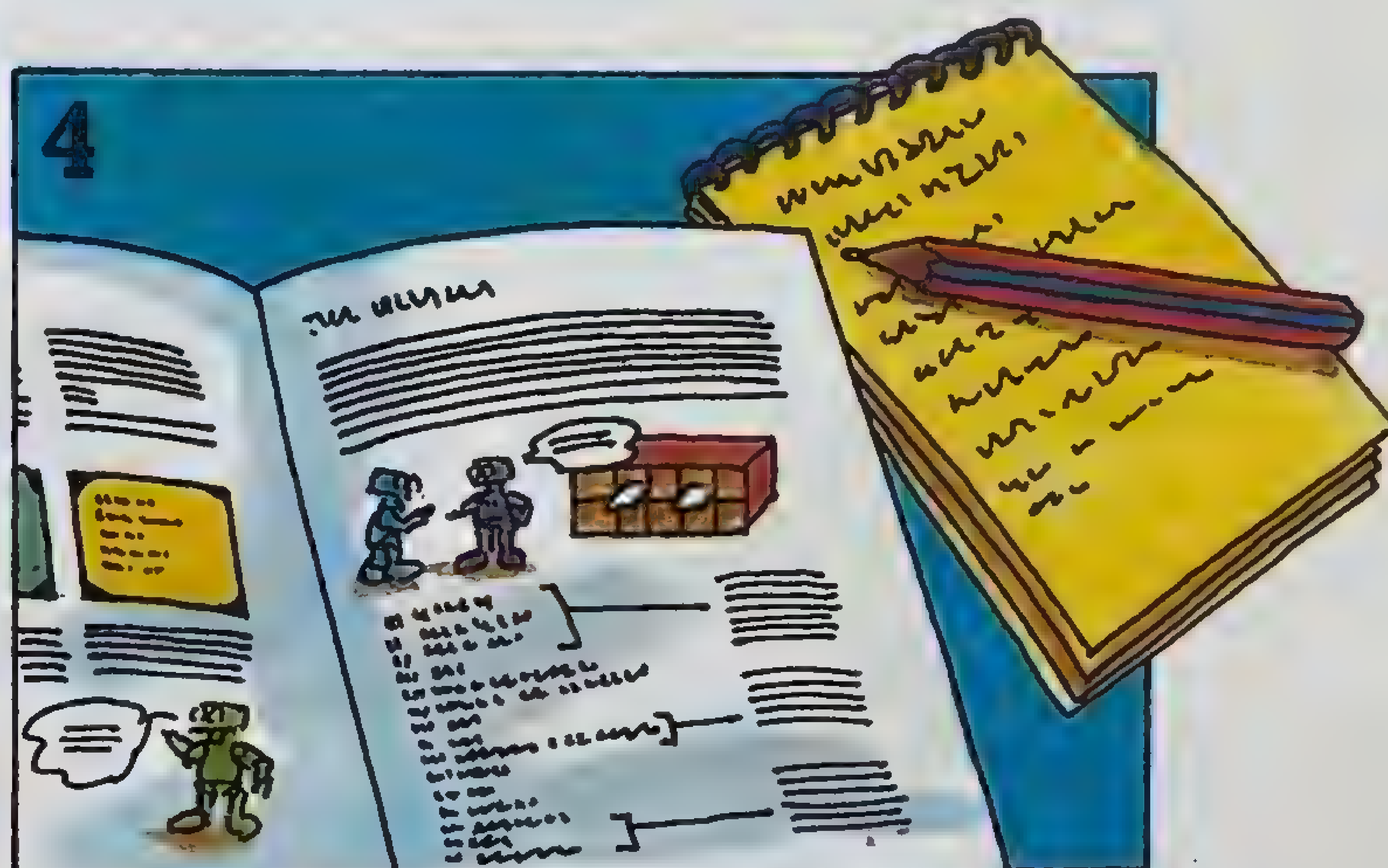
Después de averiguar para qué sirven las variables escribe el programa en una computadora. Como los programas están escritos en BASIC standard, puede que tengas que cambiar algunos de los comandos BASIC para que sirvan a tu computadora.



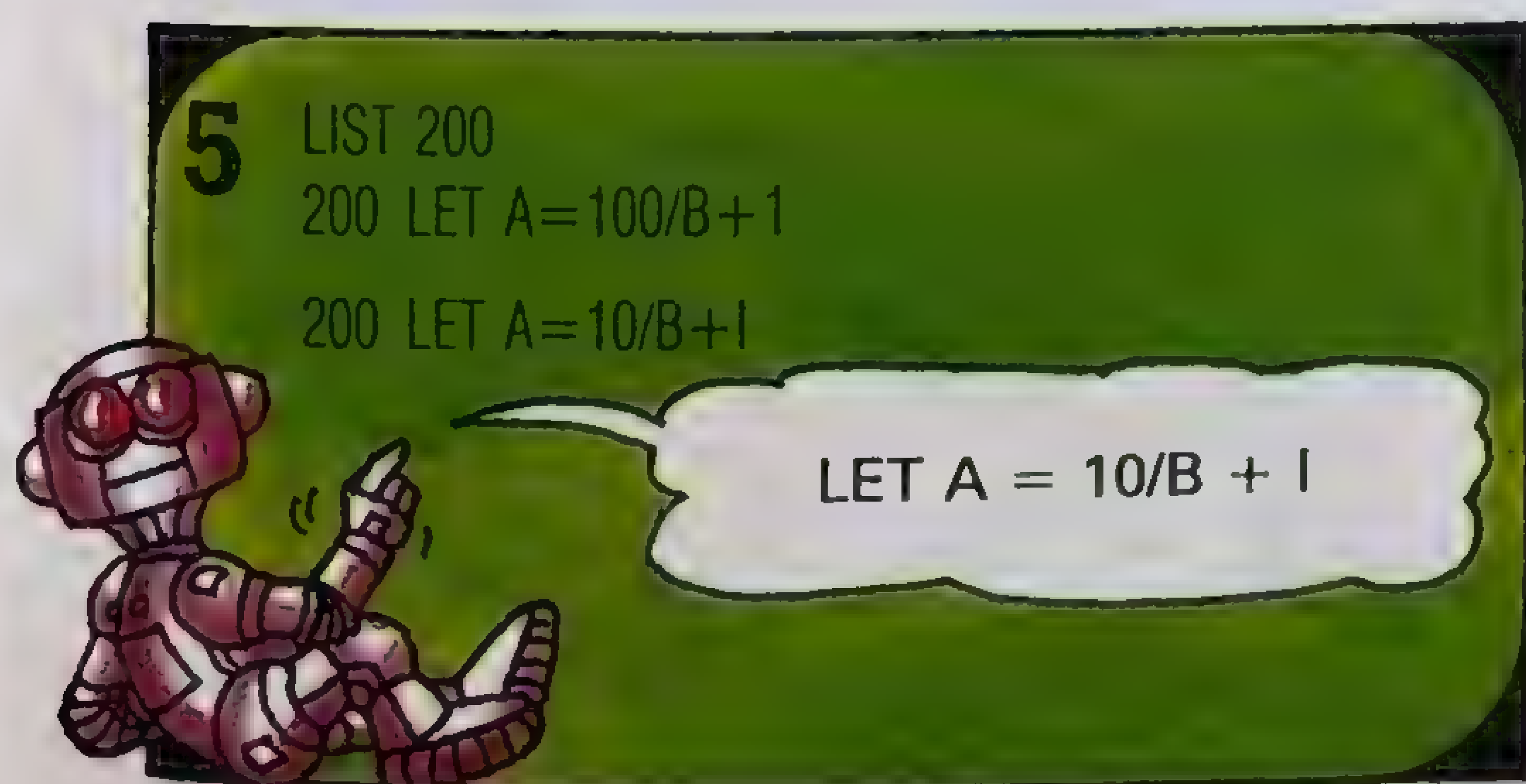
Entonces prueba y ejecuta el programa. Habrá seguramente algunos errores así que saca el listado del programa en la pantalla y busca errores tipográficos o comandos que sean incorrectos para tu computadora.



Una vez hayas encontrado todos los errores ejecuta el programa algunas veces para ver cómo funciona. Es una buena idea a estas alturas grabarlo en un cassette o disco para que no tengas que escribirlo de nuevo.



Entonces vuelve al listado y estudia cada línea intentando averiguar qué hace. Busca pequeñas rutinas que pueden servirte para incorporarlas en tus propios programas.



Puedes usar la computadora para que te ayude a comprender cómo funciona el programa. Prueba a alterar el valor de una de las variables y fíjate como afecta al programa. Haz sólo un pequeño cambio cada vez para que puedas ver el efecto de cada cambio. Recuerda escribir los valores correctos otra vez cuando hayas terminado.



También puedes insertar líneas para escribir los valores de las variables para que así puedas ver cómo cambian a lo largo del programa. Puedes encontrar útil insertar comandos STOP también y así poder estudiar el programa por etapas, pero recuerda borrarlos después. Algunas computadoras tienen un comando CONTINUE que puedes usar después de STOP.

Uso de cadenas

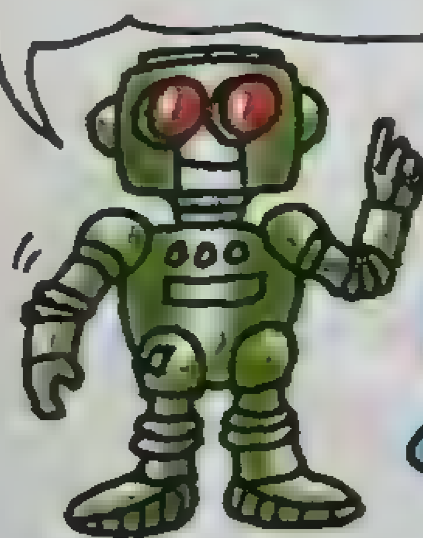
Este programa te enseña cómo cambiando comandos BASIC simples puedes hacer que la computadora lleve a cabo tareas complejas. El programa es un juego de encontrar palabras en el que la computadora te pregunta una palabra, entonces escribe las letras aleatoriamente a lo largo de la pantalla y te pregunta que aciertes el número de veces que aparece la palabra. Usa los comandos para manejar cadenas MID$, RIGHT$ y LEN y números aleatorios*.

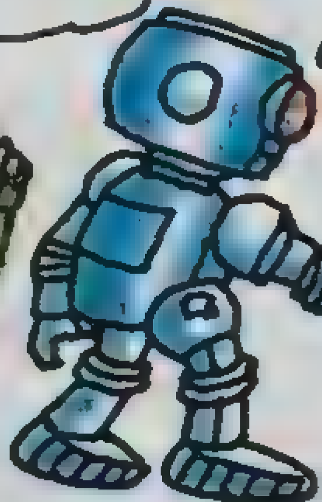
Hay dos tareas principales para llevar a cabo en el programa escribir las letras aleatoriamente en la pantalla y hacer que la computadora cuente el número de veces que la palabra aparece correctamente.

```
JUEGO DE ENCONTRAR PALABRAS
POR FAVOR, TECLEA UNA PALABRA
¿TARTA?
AHORA MIRA SI PUEDES ENCONTRAR TU PALABRA
A MEDIDA QUE LAS LETRAS APARECEN EN LA
PANTALLA
PRESIONA RETURN PARA EMPEZAR
```

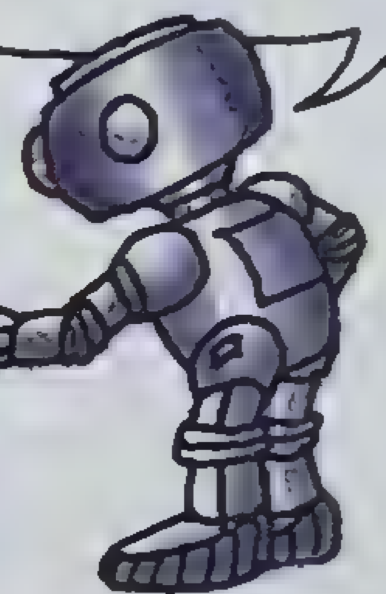
```
C K E K A E C A A K C K A E C
A K E C E A K C E A K C A K E
A C A E E C K K C C A E C A K
TECLEA EN FORMA DE NUMERO LAS VECES QUE
CREAS QUE TU PALABRA HA APARECIDO: 1
¡INCORRECTO!
TU PALABRA HA APARECIDO 2 VECES
```

Elección de las letras aleatorias







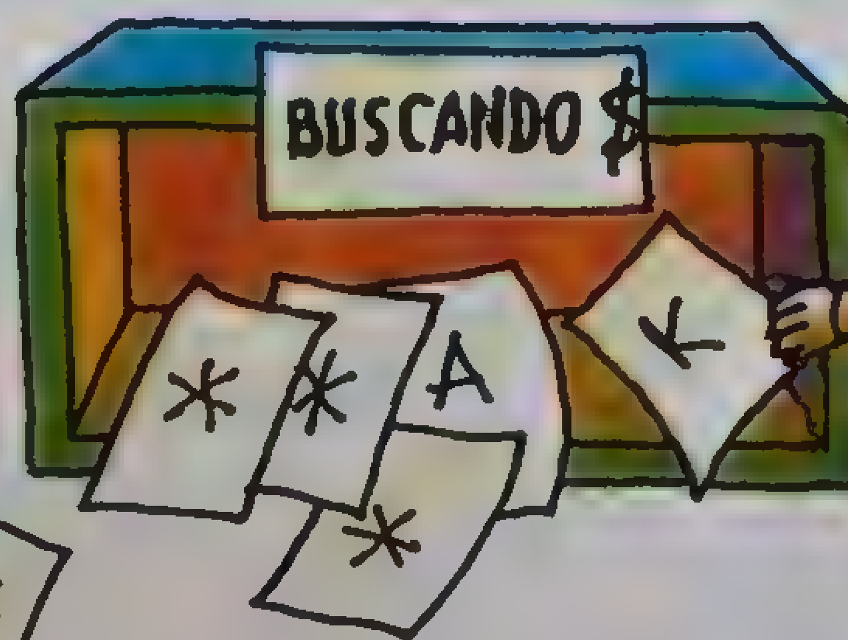


El programa usa MID$ con un número aleatorio para elegir las letras que escribe. Tu palabra está almacenada en W$. En la línea 160 toma un número aleatorio de 1 hasta la longitud de tu palabra y lo almacena en R. Entonces la línea 170 usa el número en R para decidir qué letra selecciona de W$. Guarda la letra en L$ y en la línea 180 la escribe en la pantalla. Cada vez que el bucle de la línea 150 a 200 se repite se almacena un nuevo número en R y se escoge una letra de W$.

Buscando la palabra

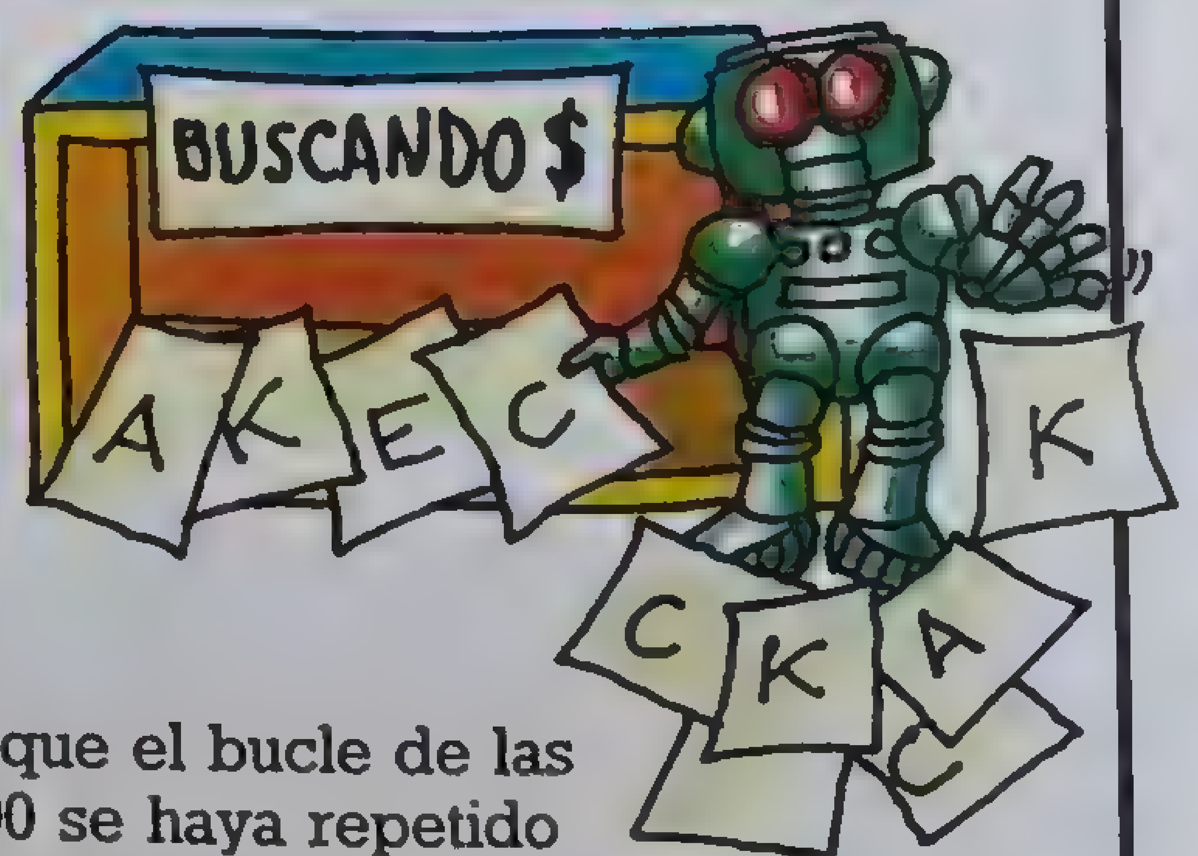


Línea 70





Línea 200



Después de que el bucle de las líneas 150-200 se haya repetido varias veces, la variable BUSCANDO$ está llena de letras.

Al principio del programa la computadora pone aparte un espacio en la memoria llamado BUSCANDO$ y lo llena con el mismo número de asteriscos que letras en tu palabra. Cada vez que coge una nueva letra aleatoria quita el primer carácter de BUSCANDO$ y pone la letra aleatoria al final de la cadena (línea 200). Entonces en la línea 20 compara BUSCANDO$ con W$ y si las letras están en el mismo orden añade 1 a N.

* Para adaptar el programa para computadoras Sinclair (Timex) ver página 46.

Juego de encontrar palabras

```
10 CLS
```

Usa el comando de tu computadora para borrar la pantalla.

```
20 PRINT «JUEGO DE ENCONTRAR PALABRAS»: PRINT
```

Esta es una línea multisentencial con un punto y coma que separa 2 instrucciones.

```
30 LET BUSCANDO$=«»
40 LET N=0
```

Inicia variables vacías para usar más tarde en el programa.

```
45 PRINT «POR FAVOR TECLEA UNA PALABRA CORTA»
50 INPUT W$
```

Te pregunta por una palabra y la pone en W$.

```
60 FOR I=1 TO LEN(W$)
70 LET BUSCANDO$=BUSCANDO$ +«*»
80 NEXT I
```

Bucle para repetirse tantas veces como letras haya en tu palabra, esto es LEN W$. Cada vez que se repite el bucle se pone un * en BUSCANDO$.

```
90 PRINT
100 PRINT «AHORA MIRA SI PUEDES ENCONTRAR»
110 PRINT «TU PALABRA A MEDIDA QUE LAS LETRAS»
115 PRINT «VAN APARECIENDO EN LA PANTALLA»
120 INPUT «PRESIONA RETURN PARA EMPEZAR»; Z$
```

La línea 120 hace que la computadora espere a que teclees algo. En la mayoría de las computadoras puedes simplemente presionar RETURN pero en el Oric, presiona una tecla y entonces pulsa RETURN.

La manera más útil de hacer esperar a la computadora hasta que estés preparado.



```
130 CLS
140 REM ELECCION DE LETRAS ALEATORIAS
150 FOR I=1 TO 50*LEN(W$)
```

Inicia un bucle desde las líneas 150 a 220 para repetirse 50 × el número de letras de tu palabra.

```
160 LET R=INT(RND(1)*LEN(W$)+1)
```

Elige un número aleatorio de hasta la longitud de tu palabra y lo pones en R.

```
170 LET L$=MID$(W$, R, 1)
```

Usa el número en R para recoger una letra de W$ y la guarda en L$.



Usa el comando RND de tu computadora.

```
180 PRINT L$+« »;
```

Escribe la letra en L$ seguida de un espacio punto y coma hace que la computadora se quede en la misma línea a escribir cada carácter.

```
190 REM BUSCANDO LA PALABRA
200 LET BUSCANDO$=RIGHT$(BUSCANDO$, LEN(W$)−1)+L$
```

Esto significa toma LEN(W$) − 1 letras de la derecha de BUSCANDO$, añade la letra en L$, y entonces pone el nuevo conjunto de letras otra vez en BUSCANDO$.

```
210 IF BUSCANDO$=W$ THEN LET N=N+1
220 NEXT I
```

N guarda el número de veces que la palabra aparece correctamente.



Esta es una manera útil de hacer que la computadora busque en su información para encontrar una palabra determinada. Puedes usar esta rutina en otros programas. También necesitas el bucle de las líneas 60 a 80.

```
230 FOR I=1 TO 1000
240 REM NO HAGAS NADA
250 NEXT I
```

Este es un bucle de retardamiento. No hay ninguna instrucción que llevar a cabo pero hace que la computadora se pare unos segundos mientras pasa por todos los valores de I.

Algunas computadoras son más rápidas que otras así que cambia el número para que corresponda a tu computadora. Un número más alto en la línea 230 provoca una pausa más larga.



```
260 CLS
265 PRINT «TECLEA EN FORMA DE NUMERO LAS VECES»
270 PRINT «CREES QUE TU PALABRA HA APARECIDO EN LA PANTALLA»
275 INPUT G
```

Guarda tu número en G.

```
280 PRINT
290 IF G=N THEN PRINT CORRECTO!»
300 IF G<>N THEN PRINT «INCORRECTO!»
310 PRINT «TU PALABRA APARECIO»; N; «VECES»
```

Compara G con N (la variable que la computadora usa para contar el número de palabras correctas).

Bucles y números aleatorios

Este programa es un juego especial que también comprueba tu cálculo mental. Enseña algunas de las memorias que hay que utilizar, bucles y números aleatorios y también hay algunos efectos especiales de pantalla que podrás usar para tus propios programas. Es un programa bastante largo pero la mayoría de las líneas son sentencias PRINT para preparar la pantalla para el juego.

ESTA ES LA COMPUTADORA DE TU NAVE HABLANDO...
TENEMOS PROBLEMAS. NO PUEDO CALCULAR LAS PROPORCIONES DE ALIMENTACION DE COMBUSTIBLE.
A MEDIDA QUE NOS APROXIMAMOS A LA TIERRA TENDRAS QUE HACER LOS CALCULOS. PUEDO DECIRTE CUANTO COMBUSTIBLE NECESITAMOS EN CADA ETAPA Y EL PERIODO DE TIEMPO EN EL QUE HA DE SER UTILIZADO. DEBES DIVIDIR EL COMBUSTIBLE ENTRE EL TIEMPO PARA DARME UNA PROPORCION CON LA QUE LA NAVE HA DE QUEMAR EL COMBUSTIBLE.

Juego de emergencia en vuelo espacial

```
100 CLS
110 FOR I=1 TO 20
115 PRINT «****ATENCION**»;
118 PRINT «**ALARMA ROJA****»;
120 FOR J=1 TO 10
125 REM NO HACER NADA
130 NEXT J
135 NEXT I
140 CLS
150 FOR I=1 TO 20
155 PRINT «****DAÑOS EN LOS CIRCUITOS****»;
160 FOR J=1 TO 10
165 REM NO HAGAS NADA
170 NEXT J
180 NEXT I
190 CLS
200 PRINT «ESTA ES LA COMPUTADORA DE TU NAVE HABLANDO...»
210 PRINT
220 PRINT «TENEMOS PROBLEMAS. NO PUEDO CALCULAR LAS PRO-
    GRAMACIONES DE ALIMENTACION DE COMBUSTIBLE».
230 PRINT
240 PRINT «A MEDIDA QUE NOS APROXIMAMOS A LA TIERRA TENDRAS
    QUE HACER LOS CALCULOS».
250 PRINT
260 PRINT «PUEDO DECIRTE EL COMBUSTIBLE QUE NECESITAMOS EN
    CADA ETAPA Y EL PERIODO DE TIEMPO EN EL QUE HA DE SER UTILI-
    ZADO».
270 PRINT
280 PRINT «DEBES DIVIDIR EL COMBUSTIBLE ENTRE EL TIEMPO
    PARA DARME UNA PROPORCION CON LA QUE LA NAVE HA DE
    QUEMAR EL COMBUSTIBLE».
290 PRINT
300 PRINT «AQUI TIENES UN EJEMPLO».
310 PRINT
320 PRINT «COMBUSTIBLE=24».
330 PRINT «TIEMPO=6».
345 PRINT
350 INPUT «POR FAVOR, DIVIDE EL COMBUSTIBLE ENTRE EL TIEMPO Y
    TECLEA LA RESPUESTA RAPIDAMENte»; RESPUESTA.
360 PRINT
370 IF RESPUESTA<>4 THEN PRINT «NO. PRUEBA DE NUEVO, TU VIDA
    DEPENDE DE ELLO»: GOTO 350
380 CLS
```

Usa la instrucción de tu computadora para borrar la pantalla.

Las líneas 110 a 135 son bucles de anidamiento. Cada vez que se llevan a cabo se escribe la alarma de las líneas 115-118, y el bucle J se repite diez veces. El bucle J es un bucle de retardamiento para hacer que la computadora se detenga un momento para que puedas leer el aviso.



Cambia los números en los bucles de retardamiento para que sirvan a tu computadora.

Las líneas 150-180 trabajan como las líneas 110-135.

La parte que sigue del programa escribe una descripción del juego en la pantalla. Si las frases son muy largas para tu pantalla inserta nuevas líneas PRINT.

PRINT sólo, deja líneas vacías.



Si tu computadora mueve el texto fuera de pantalla antes de que hayas tenido tiempo de leerlo borra algunas líneas con PRINT sólo.

Subraya las palabras en la línea inferior.

Tu respuesta se almacena en respuesta.

El GOTO sólo se lleva a cabo si la respuesta no es 4 (si tu computadora no usa líneas multisentenciales repite IF. THEN en una nueva línea con GOTO).

ATENCION
****
ALERTA ROJA
****
DAÑOS EN LOS CIRCUITOS
****

Este juego no tiene efectos gráficos ni de sonido pero puedes añadir instrucciones de tu computadora.

```
390 PRINT «DE ACUERDO. AHORA DEBES TENER TODAS LAS
    RESPUESTAS CORRECTAS»
395 PRINT «O LA NAVE SE VERA DAÑADA»
400 PRINT «SI COMETES MAS DE DOS ERRORES»
405 PRINT «TODOS SEREMOS DESTRUIDOS»
410 PRINT
420 INPUT «PRESIONA RETURN PARA DECIRME QUE
    ESTAS LISTO»; Z$
430 CLS
440 PRINT «¡¡¡ATENTO, ESTAMOS EMPEZANDO!!!»
450 LET «DAÑOS=0»
460 FOR FUEL=720 TO 120 STEP−120
470 LET T=INT (RND(1)*5+2)
480 PRINT
490 PRINT «COMBUSTIBLE=»; COMBUSTIBLE
500 PRINT «TIEMPO=»; T
510 PRINT
520 LET R=COMBUSTIBLE/T
530 INPUT «DAME LA PROPORCION AHORA»; RESPUESTA
540 IF RESPUESTA=R THEN GOTO 600
550 LET DAÑOS=DAÑOS+1
560 PRINT
570 PRINT «****DAÑOS****»
580 PRINT
590 IF DAÑOS>2 THEN GOTO 640
600 NEXT FUEL
610 CLS
620 PRINT «FELICIDADES — LO HAS HECHO TAN BIEN COMO YO
    HABRIA PODIDO HACERLO. HAS ALCANZADO LA TIERRA SIN
    DAÑOS»
630 GOTO 720
640 CLS
650 FOR I=1 TO 20
660 PRINT «*»;
670 FOR J=1 TO INT (RND(1)*I+50)
680 PRINT «      »;
690 NEXT J
700 NEXT I
710 PRINT: PRINT «LA NAVE HA SIDO DESTRUIDA»
720 END
```

Líneas 420: Espera a que presiones RETURN. (para la computadora ORIC presiona una tecla y después RETURN).

Línea 450: Inicia una variable llamada Daños.

Línea 460: Combustible es una variable para contar los bucles. También se usa en los cálculos. En la línea 460, combustible = 720 y decrece en 120 cada vez que se repite el bucle. Estos números se escogerán de forma que la suma en la línea 520 es siempre número entero.

Línea 470: Genera números aleatorios de 2 a 6 que dividirán exactamente el valor de combustible.

Líneas 520–540: La computadora hace la operación usando los valores de Combustible y T y almacena la respuesta en R. Si das la respuesta correcta la computadora va a la línea 600.

Línea 550: La variable Daños guarda la cuenta de tus errores.

Línea 590: Si cometes más de dos errores la computadora se sale del bucle y va a la línea 640.

Línea 620: Solamente dibuja esta línea si haces menos de dos errores.

Líneas 650–700: Estos bucles de anidamiento dibujan una agrupación de estrellas aleatoria en la pantalla. Cada vez que el bucle I se repite la computadora dibuja una estrella y entonces el bucle I hace que deje un número aleatorio de espacios.

Crear un banco de datos sobre fútbol

Una base de datos es una gran cantidad de información almacenada en una computadora. La información está organizada de forma que la computadora puede comparar y combinar datos y números de varias maneras diferentes para que una persona que esté usando la base de datos pueda recibir información útil en un corto espacio de tiempo.

En las páginas siguientes hay un programa para una base de datos del campeonato del mundo de fútbol. Este es un ejemplo de una pequeña base de datos que puedes usar para averiguar qué equipo ganó la copa cada año desde 1930, o en qué año ganó un equipo concreto. Al final del programa hay algunas ideas para reformar la base de datos para que contenga información diferente como el índice de una revista o los datos de un estudio de la naturaleza.

En una base de datos hay tres partes principales. Necesitas un modo adecuado de almacenar información, un medio para recuperarla y un menú. Un menú es una lista de las diversas cosas que un programa puede hacer del que puedes escoger lo que quieras. El programa también debe ser «amistoso para el usuario» esto es, debe dar a la persona que está usando la base de datos instrucciones varias y no dejar de funcionar (crash) si se comete un error.

Ejemplos del trabajo de una base de datos

```
POR FAVOR, TECLEA EL NOMBRE DEL EQUIPO, O
TECLEA MENU PARA VER LA LISTA OTRA VEZ.
ALEMANIA DEL OESTE

ALEMANIA DEL OESTE
GANO EL CAMPEONATO DEL MUNDO EN 1954
FUE FINALISTA EN 1966
GANO EL CAMPEONATO DEL MUNDO EN 1974
FUE UN FINALISTA EN 1982

PRESIONA RETURN PARA MENU
```

```
POR FAVOR, TECLEA EL AÑO, O TECLEA MENU PARA
VER LA LISTA OTRA VEZ. 1938

EN 1938
ITALIA GANO EL CAMPEONATO

PRESIONA RETURN PARA MENU
```

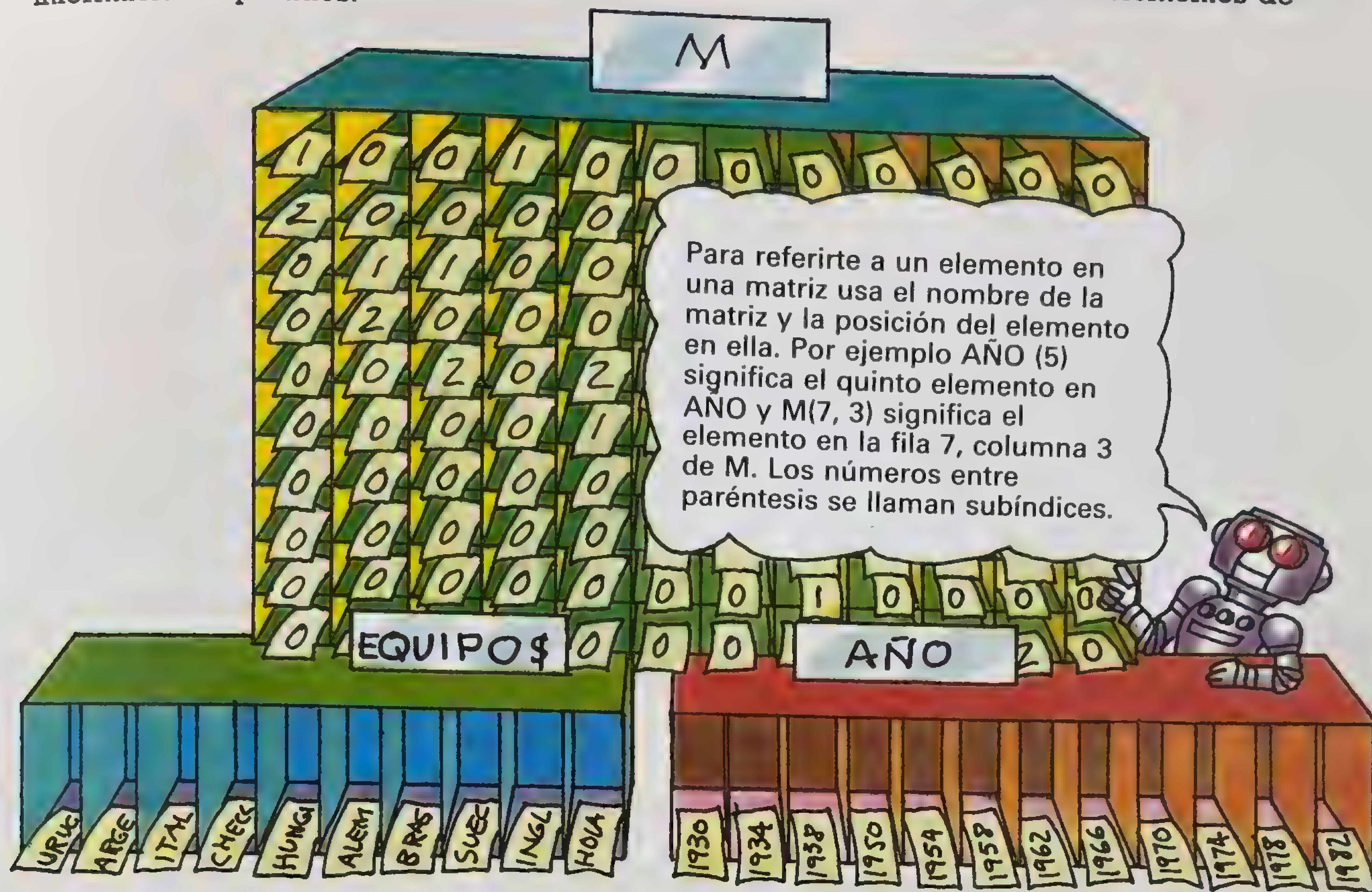
Almacenamiento de la información

Año	1930	1934	1938	1950	1954	1958	1962	1966	1970	1974	1978	1982
URUGUAY	1	0	0	1	0	0	0	0	0	0	0	0
ARGENTINA	2	0	0	0	0	0	0	0	0	0	1	0
ITALIA	0	1	1	0	0	0	0	0	2	0	0	1
CHECOSLOVAQUIA	0	2	0	0	0	0	2	0	0	0	0	0
HUNGRIA	0	0	2	0	2	0	0	0	0	0	0	0
ALEMANIA-OESTE	0	0	0	0	1	0	0	2	0	1	0	2
BRASIL	0	0	0	0	0	1	1	0	1	0	0	0
SUECIA	0	0	0	0	0	2	0	0	0	0	0	0
INGLATERRA	0	0	0	0	0	0	0	1	0	0	0	0
HOLANDA	0	0	0	0	0	0	0	0	0	2	2	0

Para emparejar los equipos y los años el programa usa una matriz de información y busca un equipo o un año igual que tú lo harías en este cuadro. En el cuadro el número 1 indica que el equipo ganó el campeonato y 2 indica que fue finalista. Leyendo a lo largo de las filas y hasta abajo en las columnas puedes ver qué equipo ganó y en qué año. El programa es una forma automática de hacer esto y desde luego con grandes cantidades de información es más rápido que un cuadro.

Construcción de una matriz

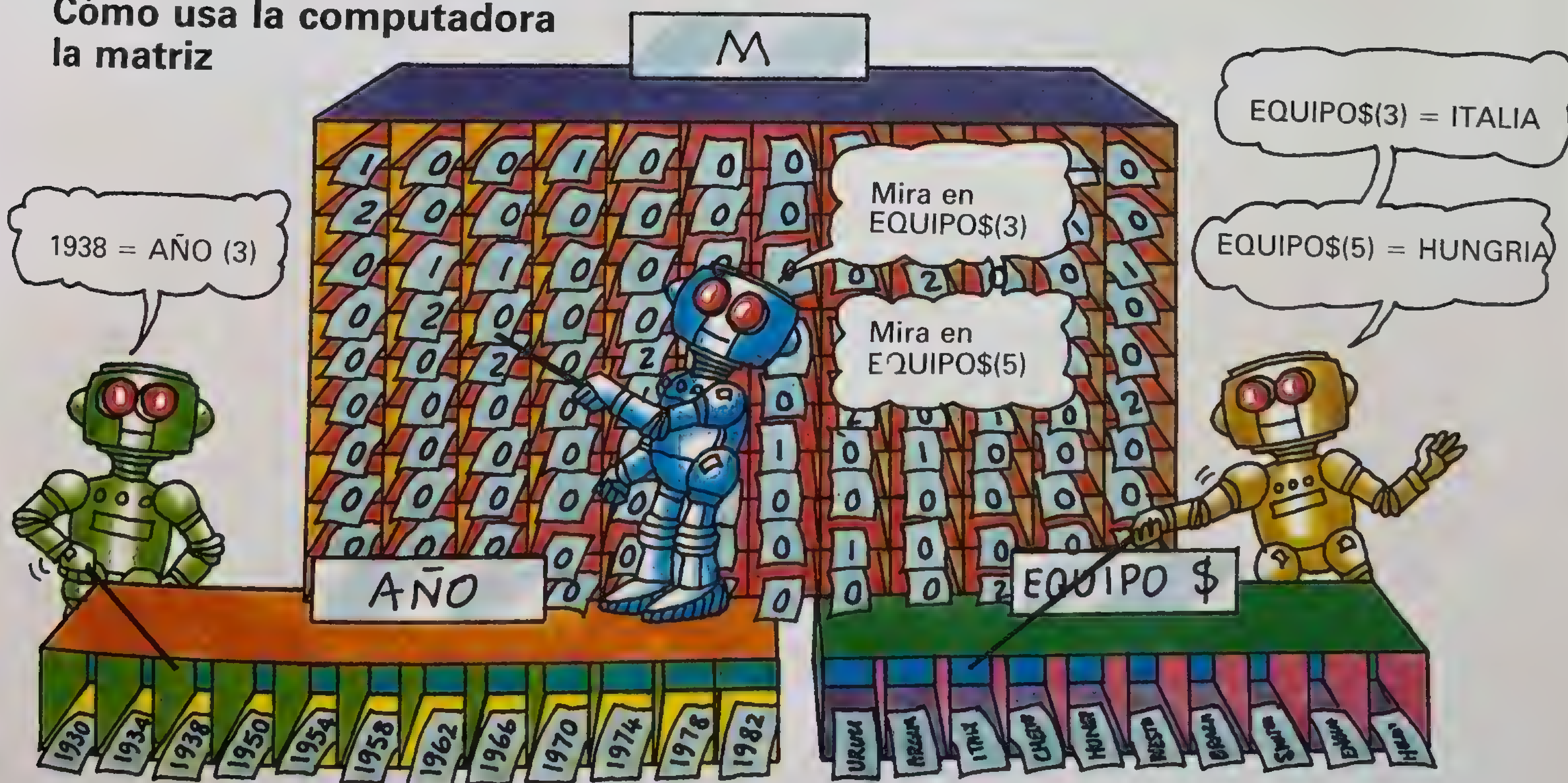
Es bastante fácil hacer una versión en computadora del cuadro de la página de al lado usando matrices. Una matriz es una variable que puede contener gran cantidad de elementos de información separados.



Necesitas dos matrices unidimensionales, una con 12 compartimentos para contener la lista de los años y otra con 10 para contener los equipos. Estas se llaman AÑO y EQUIPO $.

Para almacenar todos los datos numéricos para el cuadro necesitas una matriz bidimensional con 10 filas y 12 columnas. Esta se llama M (de Matriz) en el programa.

Cómo usa la computadora la matriz



Para encontrar qué equipo ganó el campeonato del mundo en, digamos, 1938, la computadora busca 1938 en la matriz AÑO y encuentra que está en el compartimento 3.

Entonces mira en la columna 3 de la matriz cuando encuentra un 1 ó 2 mira el número de la fila y lo usa para buscar el nombre del equipo de EQUIPO$.

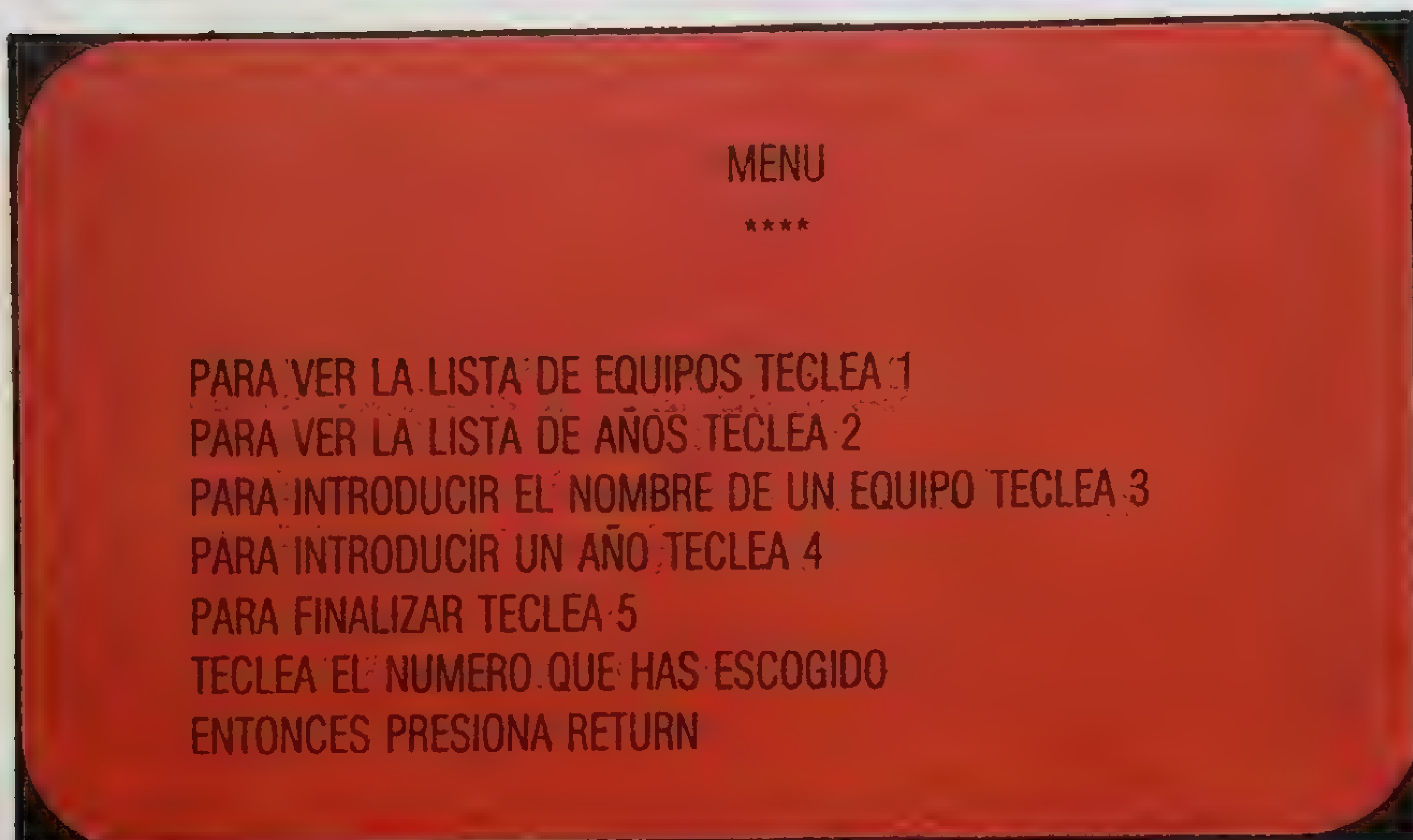
El programa de la base de datos

Hay siete partes principales dentro del programa y cada parte se lleva a cabo dentro de una subrutina diferente. Las primeras líneas le indican a la computadora qué subrutina usar y después de llevar a cabo una subrutina vuelve a estas líneas.

Las subrutinas que empiezan en las líneas 200 y 300 son para escribir la lista de equipos y años. Las líneas 400-500 son para encontrar en qué año un equipo concreto ganó la copa, mientras que las líneas 500-600 son para encontrar qué equipo ganó la copa en un año concreto. Todos los datos están listados hacia el final del programa y, seguidos por el Menú. Suele ser mejor poner los datos hacia el final del programa y dejar la parte de trabajo del programa al comienzo.

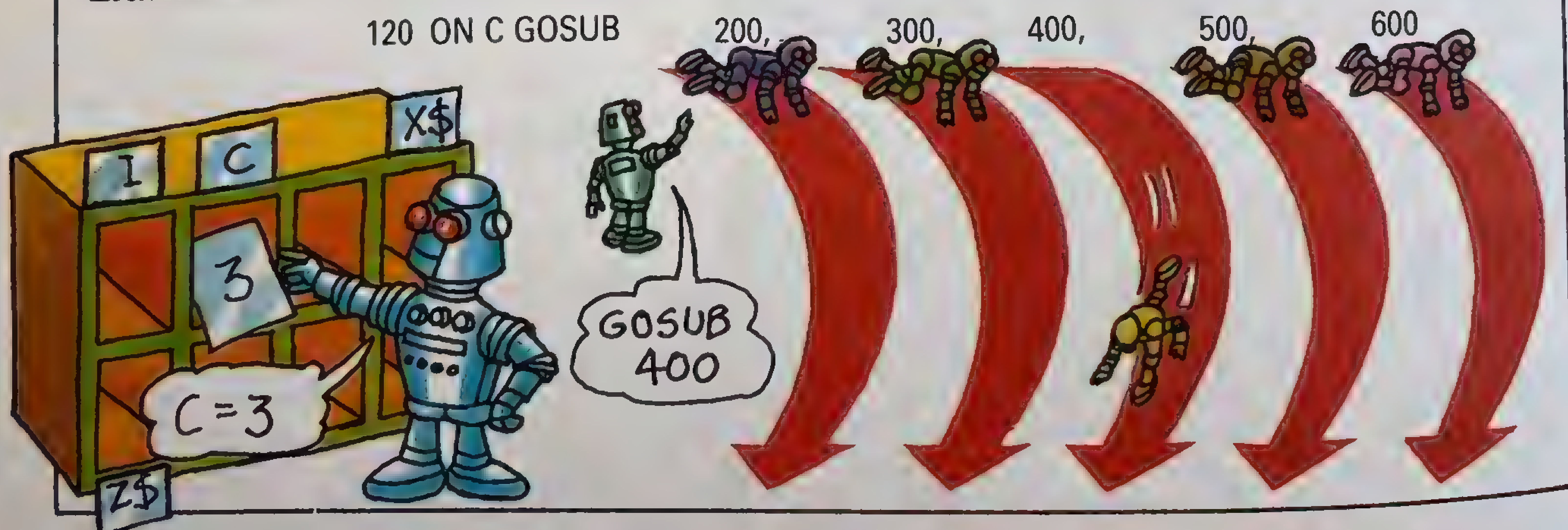


El Menú



El Menú es la parte del programa que te dice lo que el programa puede hacer y cómo usarlo. En este programa escoges lo que quieres tecleando un número. El número se guarda en una variable C y la computadora usa este número para buscar la correcta subrutina para que realice el trabajo que tu quieres.

Llamando a las subrutinas

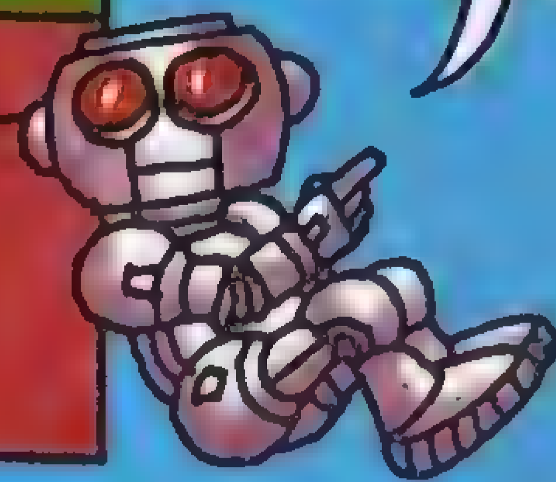


La línea 120 en el programa controla la subrutina que usa la computadora. La letra C es la variable que contiene el número que tecleaste después de ver el menú. La computadora usa el número en C para decidir a que subrutina ir. Si C = 1 va a la primera subrutina listada en la línea 120, esto es, aquella que empieza en la línea 200. Si C = 2 se va a la segunda, esto es, línea 300. Si C = 3 entonces se va a la tercera, etc. ON GOSUB es un comando BASIC muy útil para hacer que la computadora vaya a diferentes subrutinas dependiendo de un test. Si tu computadora no tiene el comando ON puedes usar varias setencias. IF... THEN en vez de ON GOSUB, ej.: IF C = 1 THEN GOSUB 200.

Para qué sirven las variables

Año	Equipo $	M
La matriz para almacenar los años.	La matriz para almacenar los nombres de los equipos	La matriz bidimensional para almacenar todos los datos numéricos.
C	**Z $**	**X $**
La variable para almacenar el número que tecleas después de ver el menú.	El nombre del equipo o año que tecleas.	Datos temporales.

Usa sólo las primeras letras si tu computadora no va a aceptar palabras como nombres de variables.



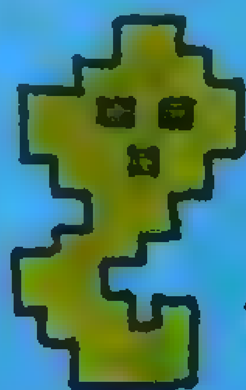
El programa*

```
10 DIM EQUIPO$(10): DIM AÑO(12): DIM M(10, 12)
100 GOSUB 1000: REM LEER DATOS
110 GOSUB 2000: REM IMPRIMIR MENU
```

Dice a la computadora cuánto espacio ha de dejar para las matrices.

Cuando ejercitas el programa la primera cosa que hace la computadora es ir a la subrutina en la línea 1000 para leer los datos.

Después, va a la línea 2000 para imprimir el menú en la pantalla.



Los paréntesis a la izquierda del listado indican las diferentes partes del programa.

```
120 ON C GOSUB 200, 300, 400, 500, 600
130 GOTO 110
```

Esta le manda a la subrutina correcta para llevar a cabo el trabajo que hayas escogido en el menú. Después de la subrutina la computadora vuelve a la línea 130 que la manda a la línea 110 para imprimir el menú otra vez.

```
200 REM SUBRUTINA PARA IMPRIMIR LA LISTA DE EQUIPOS
210 CLS
220 PRINT «LISTA DE EQUIPOS»: PRINT «-- - - - - - - - -»
230 FOR I=1 TO 10: PRINT EQUIPO$(I): NEXT I
235 PRINT
240 INPUT «PRESIONA RETURN PARA VER EL MENU»; X$
250 RETURN
```

Subraya las palabras LISTA de EQUIPOS.

Bucle para imprimir los nombres de los equipos. Cada vez que el bucle se repite I crece en 1 y la computadora imprime el nombre siguiente en EQUIPO.

Hace que la computadora espere a que presiones RETURN antes de ir a la línea siguiente (En el ORIC debes presionar una tecla).



Vuelta a la línea 130.

```
300 REM SUBRUTINA PARA IMPRIMIR LA LISTA DE AÑOS
310 CLS
320 PRINT «LISTA DE AÑOS»: PRINT «-- - - - - - -»
330 FOR I=1 TO 12: PRINT AÑO(I): NEXT I
335 PRINT
340 PRINT «NO HUBO COMPETICION NI EN 1942 NI EN 1946»
345 PRINT
350 INPUT «PRESIONA RETURN PARA VER EL MENU»; X$
360 RETURN
```

Bucle para imprimir los años.



Otra vez vuelta a la línea 130.

* Para cambiar este programa para que funcione en computadoras Sinclair (Timex) ver página 46.

```
400 REM SUBRUTINA PARA INTRODUCIR EL EQUIPO
405 CLS
410 INPUT «POR FAVOR, TECLEA EL NOMBRE DEL EQUIPO O TECLEA MENU
PARA VER LA LISTA OTRA VEZ»; Z$
```

El nombre del equipo o la palabra menú se almacena en Z$.

```
415 IF Z$=«MENU» THEN RETURN
```

Si Z$ = MENU la computadora vuelve a la línea 130 y desde ahí a la línea 110 a imprimir el Menú.

```
420 FOR I=1 TO 10
425 IF Z$=EQUIPO$(I) THEN GOTO 440
430 NEXT I
```

Bucle para comparar Z$ con cada nombre en EQUIPO$. Cuando encuentra un nombre que es igual a Z$ va a la línea 440.

```
435 PRINT: PRINT «EQUIPO NO ENCONTRADO — POR FAVOR, PRUEBE
OTRA VEZ»: PRINT: GOTO 410
```

Esta línea es la salvaguardia por si acaso cometes un error al escribir el nombre de un equipo o tecleas un nombre que no esté en la base de datos.

```
440 PRINT
445 PRINT Z$: PRINT
```

Escribe el nombre del equipo.



Si usas un micro BBC ver la nota de la página 46.

```
450 FOR J=1 TO 12
455 IF M(I, J)=1 THEN PRINT «GANO LA COPA EN»; AÑO(J): PRINT
460 IF M(I, J)=2 THEN PRINT «FUE UN FINALISTA EN»; AÑO(J): PRINT
465 NEXT J
```

Bucle para hacer que la computadora busque detalles sobre tu equipo en la matriz. I es el número de la fila. El valor de I se fija en el bucle de las líneas 420-430 y es el subíndice (el número que indica su posición en la matriz) de tu equipo en EQUIPO$. J es el número de la columna, cada vez que se repite el bucle la computadora busca en la siguiente columna a lo largo de la fila I.

```
470 PRINT
480 INPUT «PRESIONA RETURN PARA MENU»; X$
```

Lo mismo que la línea 240.

```
490 RETURN
```



Vuelta a la línea 130.

```
500 REM SUBRUTINA PARA INTRODUCIR EL AÑO
505 CLS
510 INPUT «POR FAVOR, TECLEA EL AÑO O TECLEA MENU PARA VER
LA LISTA OTRA VEZ»; Z$
```

Igual que la línea 410 pero esta vez tu año se guarda en Z$.

```
515 IF Z$=«MENU» THEN RETURN
520 FOR I=1 TO 12
525 IF VAL(Z$)=AÑO(I) THEN GOTO 540
530 NEXT I
```

Bucle para comparar Z$ con cada uno de los años en AÑO. No puedes comparar una variable de cadena con una variable numérica por lo que necesitas el comando BASIC VAL. Esto le dice a la computadora que tome los valores de Z$ como números.

```
535 PRINT: PRINT «AÑO NO ENCONTRADO — PRUEBA OTRA VEZ»:
PRINT: GOTO 510
```

```
540 PRINT: PRINT «EN»; Z$: PRINT
550 FOR J=1 TO 10
555 IF M(J, I)=1 THEN PRINT EQUIPO$(J); «GANO LA COPA»
560 NEXT J
565 PRINT
570 INPUT «PRESIONA RETURN PARA EL MENU»; X$
580 RETURN
```

Este bucle trabaja de la misma manera que las líneas 450-460. Esta vez el número de la columna fija el subíndice del año en AÑO y el número de la fila cambia cada vez que se repite el bucle.



Vuelta a la línea 130.

```
600 REM SUBRUTINA PARA FINALIZAR EL PROGRAMA
610 INPUT «FIN-SEGURO (Y/N)»; X$
620 IF X$<>«Y» THEN RETURN ELSE END
```

Comprobaciones para estar seguro de que quieres acabar. Si tecleas el comando BASIC END le dice a la computadora que deje de ejecutar el programa. Si tecleas alguna otra cosa vuelve a la línea 130. La palabra ELSE es una manera útil de añadir más condiciones a sentencias IF...THEN. Para saber más sobre esto mira a la vuelta de la página.



Si tu computadora no usa ELSE puedes poner la palabra END en una nueva línea sola.

```
1000 FOR I=1 TO 12: READ AÑO(I): NEXT I
1010 DATA 1930, 1934, 1938, 1950
1020 DATA 1954, 1958, 1962, 1966
1030 DATA 1970, 1974, 1978, 1982
```

Bucle para introducir los datos dentro del AÑO.

```
1100 FOR I=1 TO 10: READ EQUIPO$(I): NEXT I
1110 DATA URUGUAY, ARGENTINA
1115 DATA ITALIA, CHECOSLOVAQUIA, HUNGRIA
1120 DATA ALEMANIA DEL OESTE, BRASIL
1125 DATA SUECIA, INGLATERRA, HOLANDA
```

Bucle para introducir datos en EQUIPO$.





Ten mucho cuidado al teclear estos datos. Si te dejas alguna coma o carácter obtendrás un error.

```
1200 FOR I=1 TO 10: FOR J=1 TO 12
1205 READ M(I, J)
1210 NEXT J: NEXT I
1215 RETURN
```

Bucles de anidamiento para introducir información en la matriz bidimensional M.



Vuelta a la línea 110 esta vez.

```
1220 DATA 1,0,0,1,0,0,0,0,0,0,0,0
1230 DATA 2,0,0,0,0,0,0,0,0,0,1.0
1240 DATA 0,1,1,0,0,0,0,0,2,0,0,1
1250 DATA 0,2,0,0,0,0,2,0,0,0,0,0
1260 DATA 0,0,2,0,2,0,0,0,0,0,0,0
1270 DATA 0,0,0,0,1,0,0,2,0,1,0,2
1280 DATA 0,0,0,0,0,1,1,0,1,0,0,0
1290 DATA 0,0,0,0,0,2,0,0,0,0,0,0
1300 DATA 0,0,0,0,0,0,0,1,0,0,0,0
1310 DATA 0,0,0,0,0,0,0,0,0,2,2,0
```

Esta es la información para M.

Es una buena idea comprobar la información varias veces leyendo a lo largo de las filas y para abajo en las columnas. Si algún número está mal la computadora te dará información incorrecta cuando mire en la matriz.

EL LISTADO CONTINUA A LA VUELTA DE LA PAGINA



```
2000 REM SUBRUTINA PARA IMPRIMIR EL MENU
2010 CLS
2020 PRINT «                              MENU»
2030 PRINT «                              *****»
2040 FOR I=1 TO 6: PRINT: NEXT I
2050 PRINT «PARA VER LA LISTA DE EQUIPOS TECLEA 1»
2060 PRINT
2070 PRINT «PARA VER LA LISTA DE AÑOS TECLEA 2»
2080 PRINT
2090 PRINT «PARA INTRODUCIR EL NOMBRE DE UN EQUIPO TECLEA 3»
2100 PRINT
2110 PRINT «PARA INTRODUCIR UN AÑO TECLEA 4»
2120 PRINT
2130 PRINT«PARA FINALIZAR TECLEA 5»
2140 PRINT
2150 PRINT «TECLEA EL NUMERO DE TU OPCION»
2160 INPUT «ENTONCES PRESIONA RETURN»; C
2170 IF C<1 O C>5 THEN PRINT «POR FAVOR, INTRODUCE UN NUMERO EN-
     TRE 1 Y 5»: GOTO 2150
2180 RETURN
```

Deja como 15 espacios aquí para centrar la palabra Menú por encima de la lista de opciones.

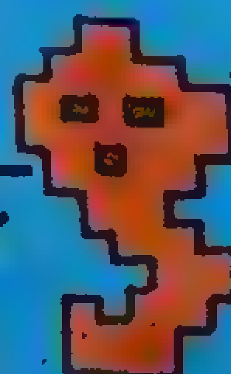
Bucle para dejar 6 líneas en blanco.



Estas líneas imprimen menú. Un menú debe ser claro y amistoso para el usuario para que la persona que esté utilizando el programa sepa exactamente qué hacer.

El número de tu opción se almacena en C.

Esta línea es una salvaguardia en caso de que teclees un número que no esté entre 1 y 5. Para más sobre OR ver abajo.



Vuelta a la línea 120 para seleccionar la subrutina correcta.

AND, OR y ELSE*

```
IF A=3 AND C$=«SI» THEN LET D=D+1

IF X<0 OR X>100 THEN PRINT «FUERA DE RANGO»

IF AGE<36 THEN PRINT «JOVEN» ELSE
PRINT «VIEJO»
```

Puedes usar estas palabras BASIC para añadir unos tests e instrucciones a sentencias IF...THEN, como se muestra en los ejemplos de arriba. Cuando usas AND la computadora llevará a cabo la instrucción de THEN sólo si los dos tests en la sentencia IF son verdaderos OR le dice que lleve a cabo la instrucción si alguno es verdadero. La palabra ELSE te permite hacer que la computadora lleve a cabo instrucciones si ninguno de los tests son verdaderos. Puedes escribir algún programa corto para resolver el problema de la derecha.

* Estas palabras BASIC están disponibles en todas las computadoras.

Problema



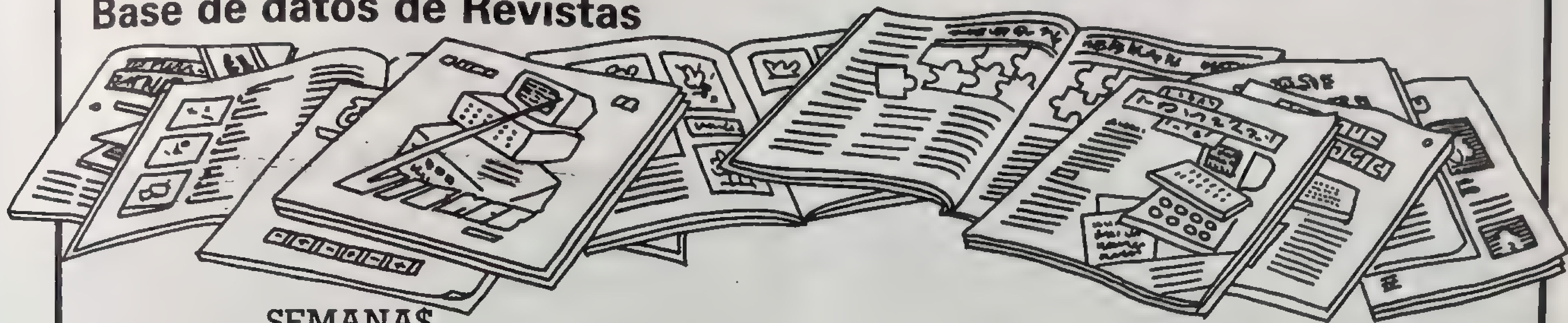
El robot corredor campeón Zak, puede correr 500 metros por segundo. Pero si la temperatura sube por encima o baja por debajo de 60° Zak acelera o desacelera 10 metros por segundo. Puedes escribir un programa para imprimir la distancia que puede recorrer Zak de acuerdo con la temperatura y número de segundos que introduzcas (Respuesta página 48).

Transformación de la base de datos

Una vez que hayas comprendido cómo funciona el programa es bastante fácil transformarlo para hacer una base de datos de un tema diferente. Abajo hay algunas ideas para diferentes bases de datos.

Cuando hayas decidido un tema para la base de datos, dibuja un cuadro con todos los números para la información como el de la página 18. Tu cuadro puede tener un número diferente de filas y columnas en cuyo caso tendrás que cambiar el tamaño de las matrices en el programa. A continuación pon toda tu información en las líneas de datos en el programa y reescribe, también, las preguntas del menú. Recuerda cambiar las sentencias DIM y el número de veces que se repiten los bucles para introducir la información en las matrices.

Base de datos de Revistas



SEMANA$

TEMA$

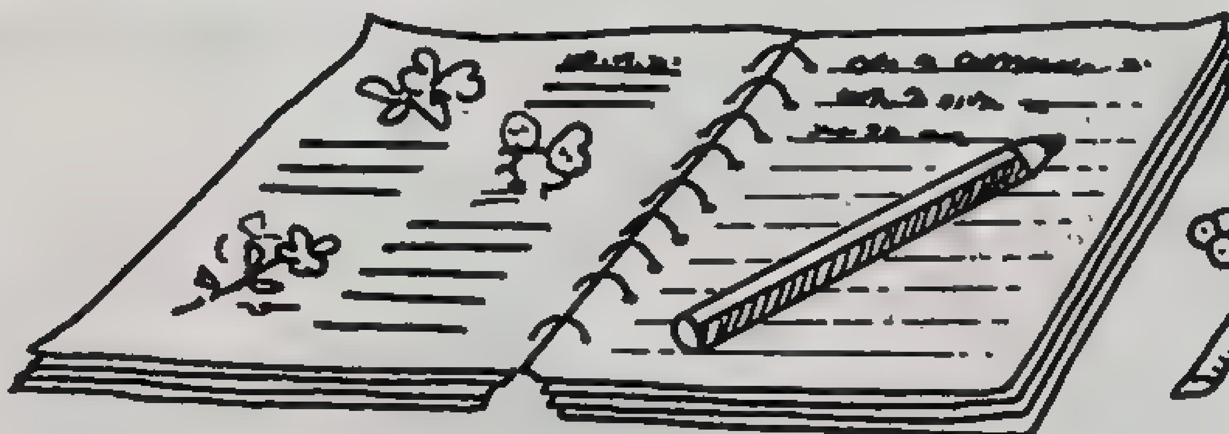
MATRIZ

Podrías usar una base de datos de revistas para ver en que mes apareció un tema o qué temas fueron cubiertos en un mes cualquiera. Esto podría ahorrarte horas ojeando revistas o buscando un artículo concreto. Necesitarás una matriz llamada SEMANA$ y otra llamada TEMA$ además de la matriz principal.

Base de datos sobre vida salvaje









PAJARO$

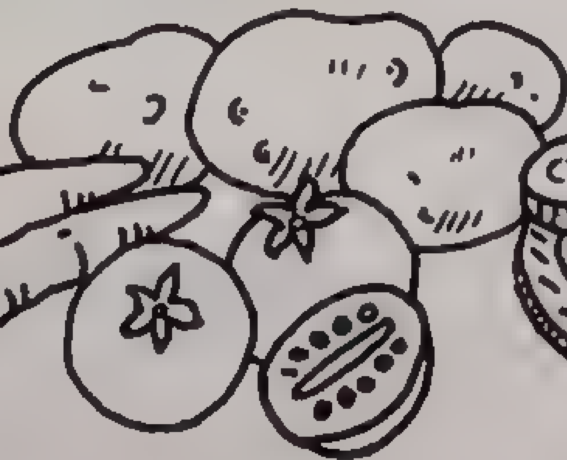
LUGAR$

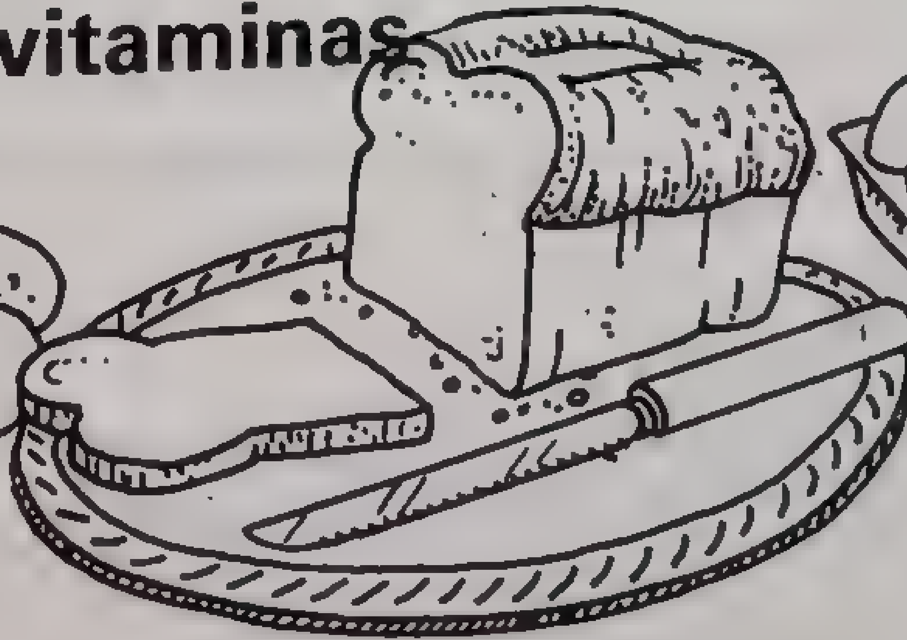
MATRIZ

Una base de datos sobre pájaros o plantas podría mostrar cuándo o dónde los viste. Necesitarás una matriz para los nombres de pájaros o plantas y otra para el lugar o época del año en que los viste. Organizaciones para la vida salvaje están recopilando bases de datos como esta para tener en cuenta la distribución de las diferentes especies.

Base de datos de comidas y vitaminas









COMIDA$

VITAMINA$

MATRIZ

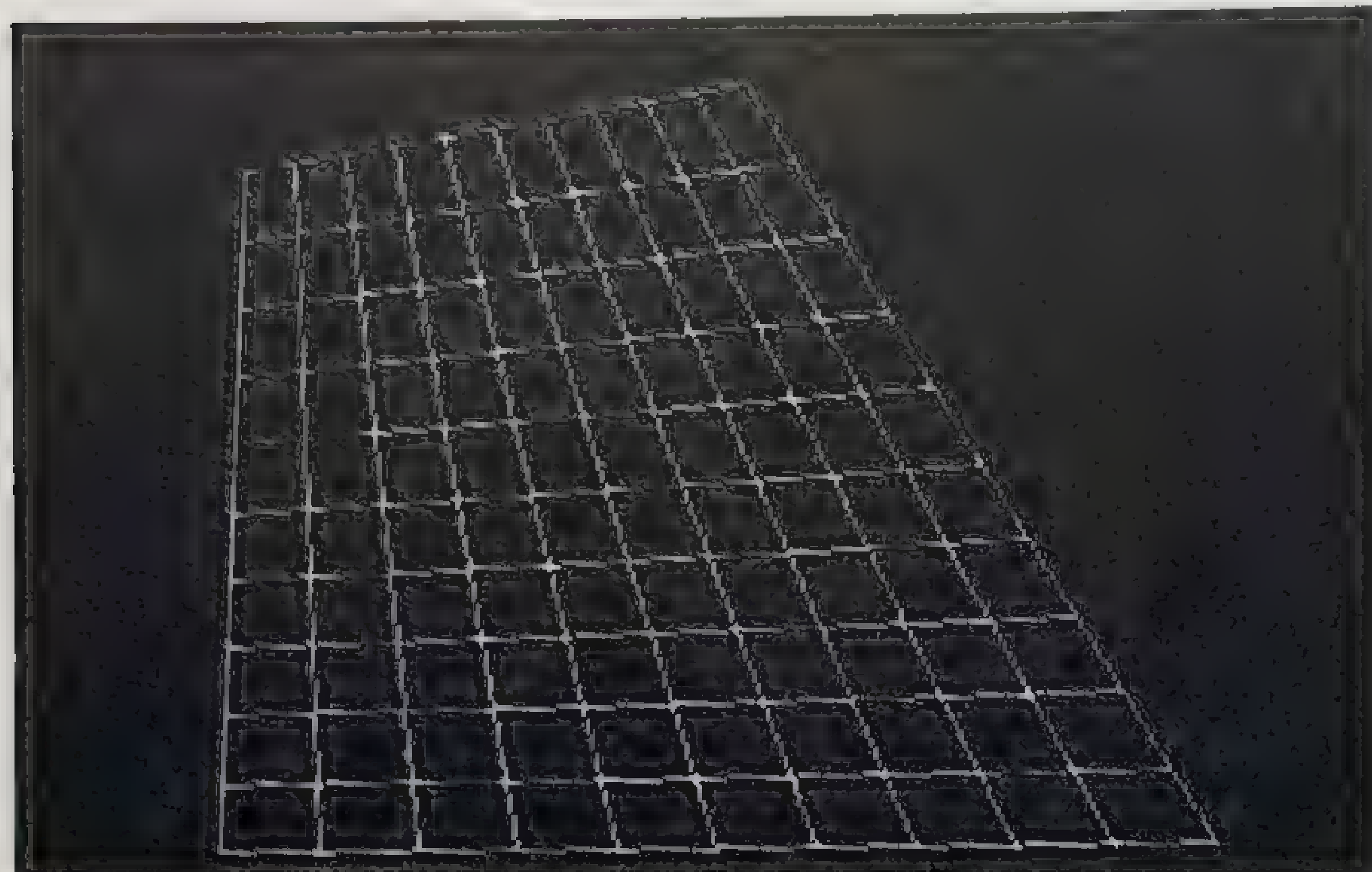
Esta base de datos te permitirá encontrar qué comidas contienen una vitamina concreta o qué vitaminas están presentes en una comida concreta. Otra idea sería una base de datos de comidas y calorías, para que puedas ver cuántas calorías hay en una comida concreta o qué comidas contienen más de un cierto número de calorías.

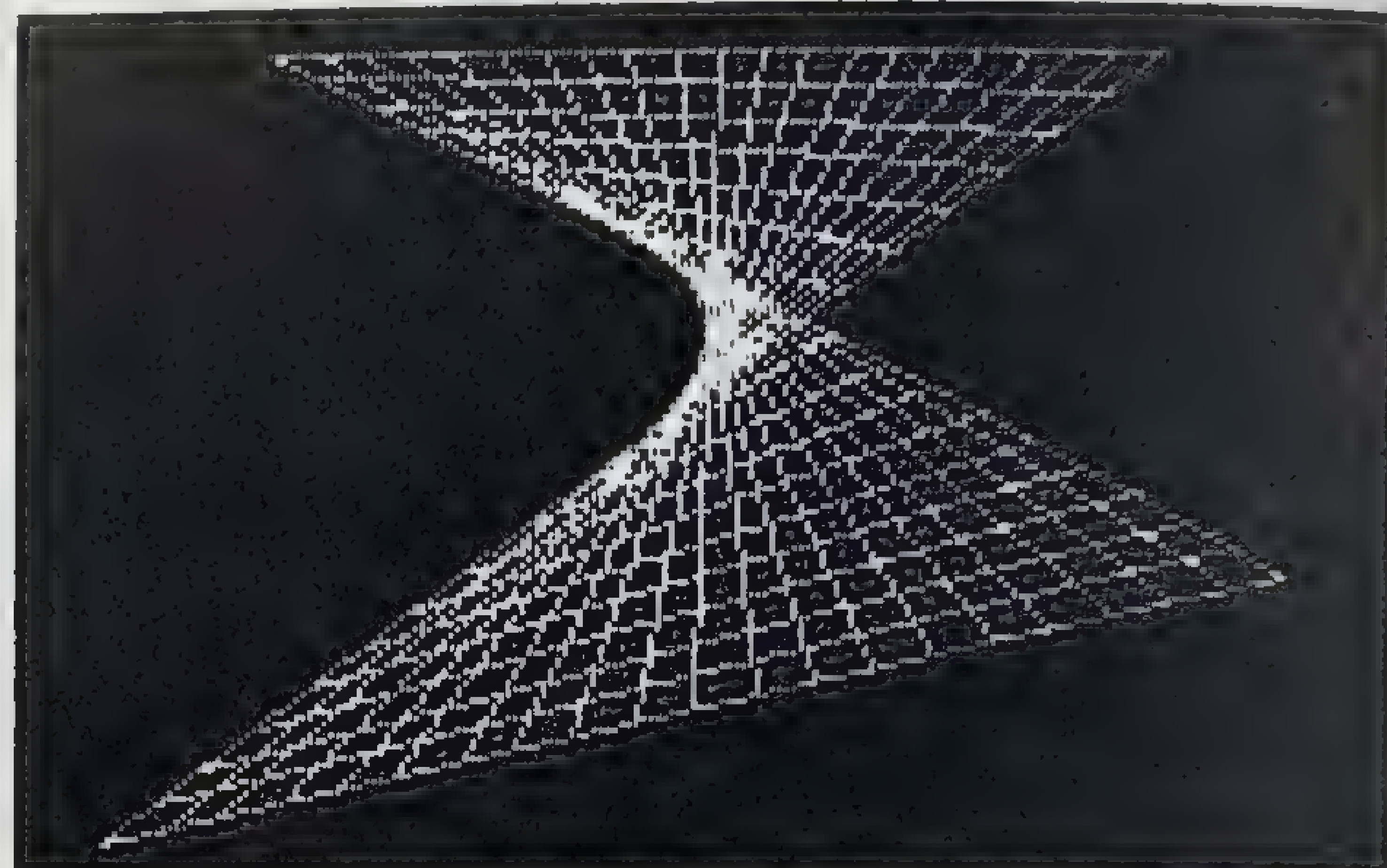
Gráficos instantáneos

Este programa dibuja formas simples y las rellena con redes de líneas cruzadas. Las redes se usan frecuentemente en gráficos de computadoras para ayudar a que las formas parezcan más tridimensionales o para darles una imagen de era espacial.

El programa usa los comandos gráficos PLOT X, Y para dibujar un punto y DRAW X, Y para dibujar una línea. Las coordenadas X e Y se miden desde el borde de la pantalla. Tendrás que transformar estas instrucciones para tu computadora y añadir los comandos de gráficos especiales que pueda necesitar tu computadora.

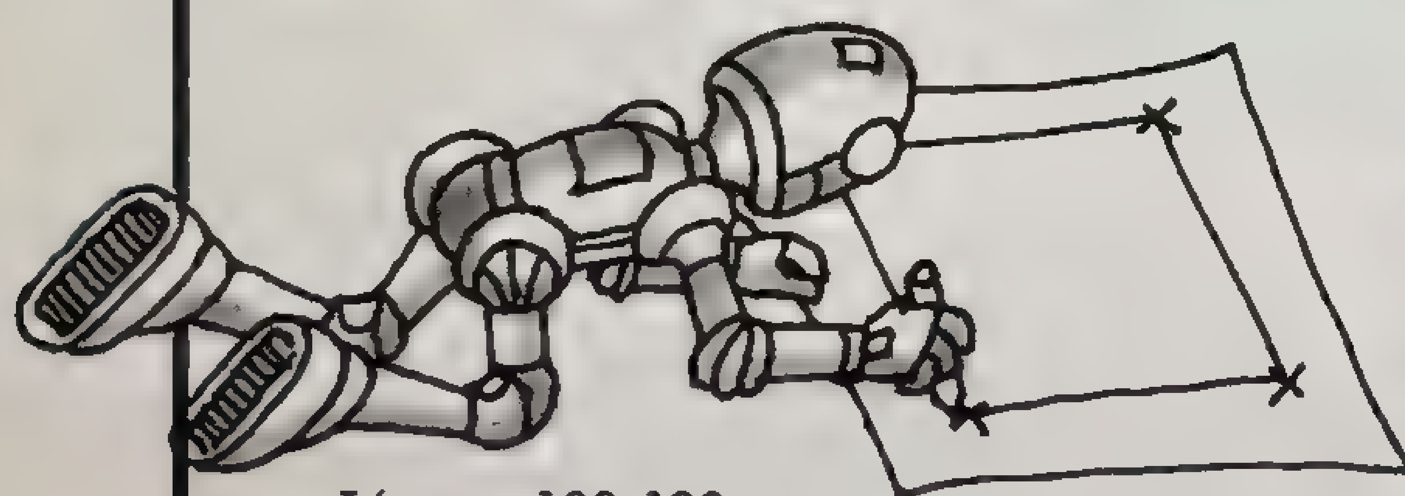
Hay dos modos de escribir programas de gráficos. Puedes decirle a la computadora que calcule y dibuje todos los puntos a medida que trabaja y construir el dibujo gradualmente en la pantalla. O puedes hacer que haga todos los cálculos antes, que los guarde en las matrices y que dibuje todo el gráfico prácticamente instantáneamente. El programa siguiente usa el método de gráficas instantáneas.



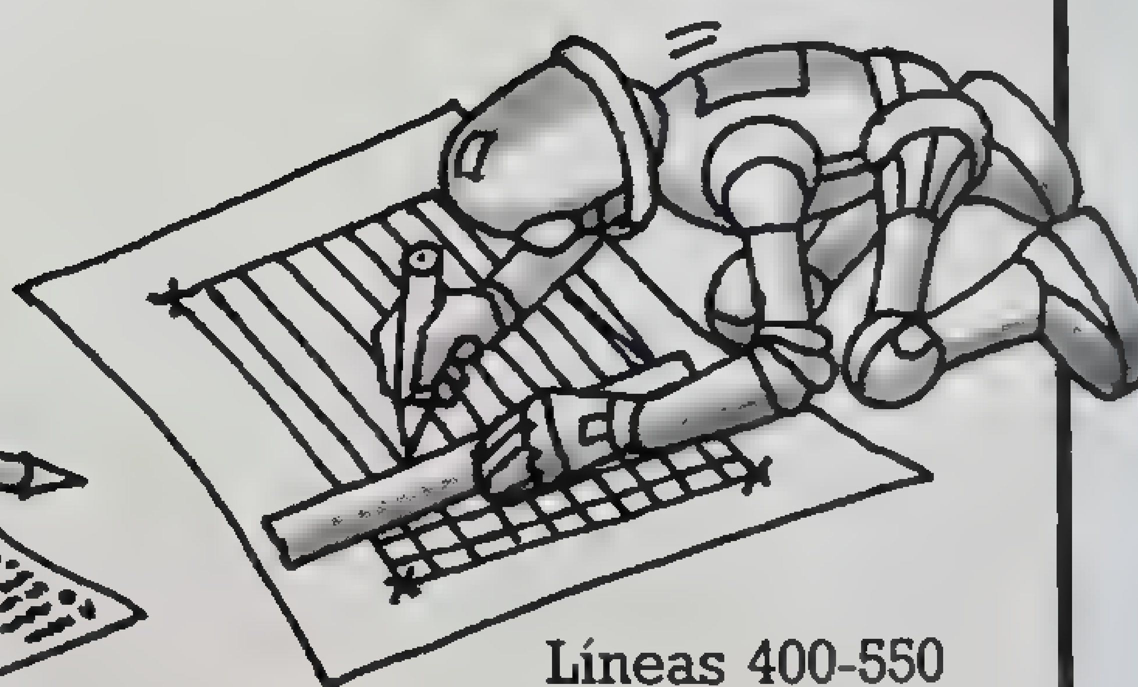


Puedes hacer todo tipo de formas cambiando los datos en el programa. También puedes dibujar las líneas de las redes en diferentes colores. Hay algunas ideas para adaptar el programa al final del listado.

Partes del programa

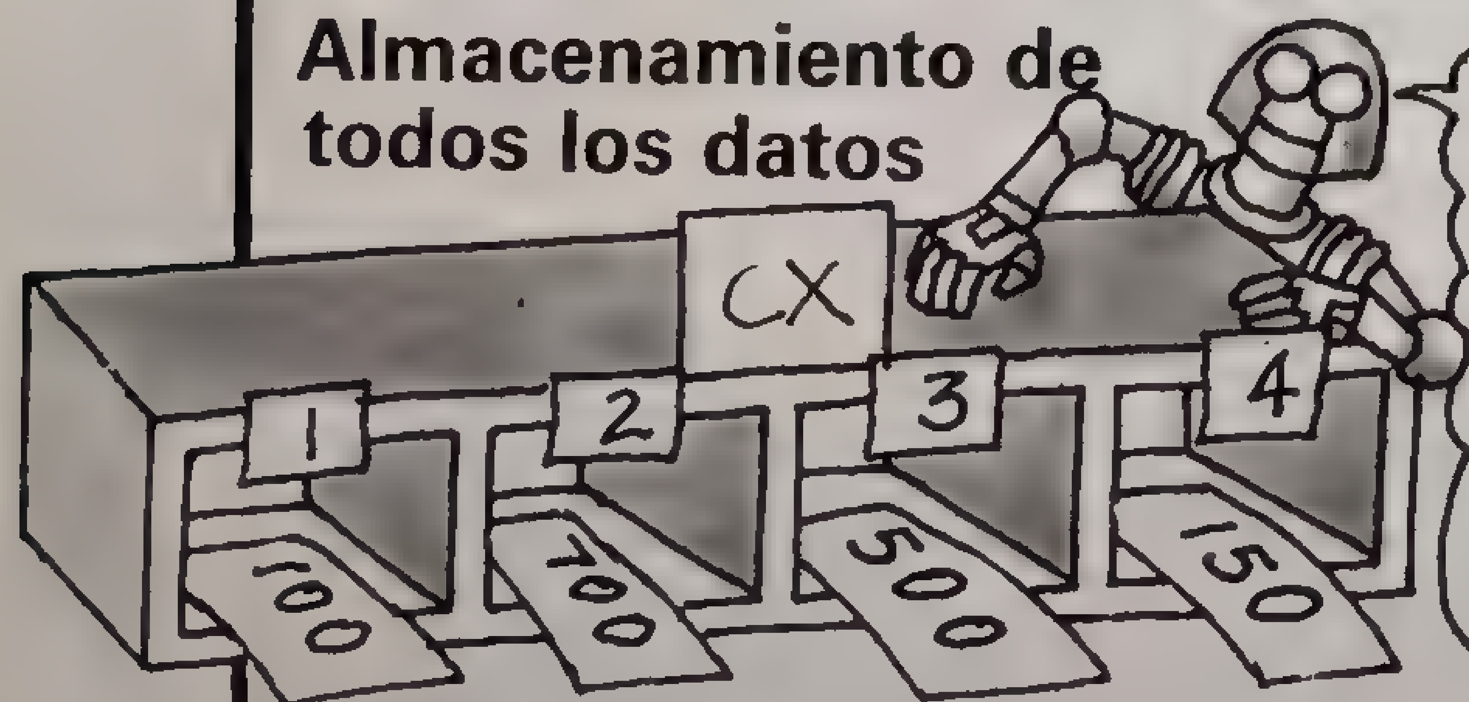




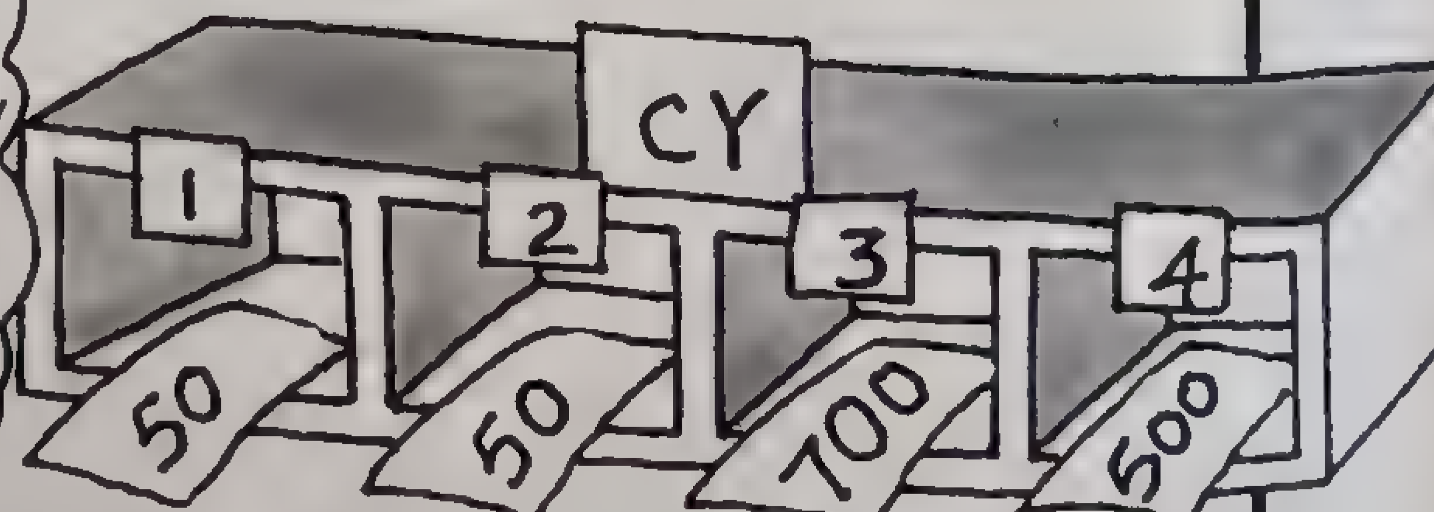


Hay tres partes principales en el programa. La primera parte (líneas 100-190) dibuja las esquinas de la red y dibuja las líneas entre ellas. La segunda parte (líneas 200-360) calcula las coordenadas para la red y la tercera parte (líneas 400-550) dibuja las líneas.

Almacenamiento de todos los datos

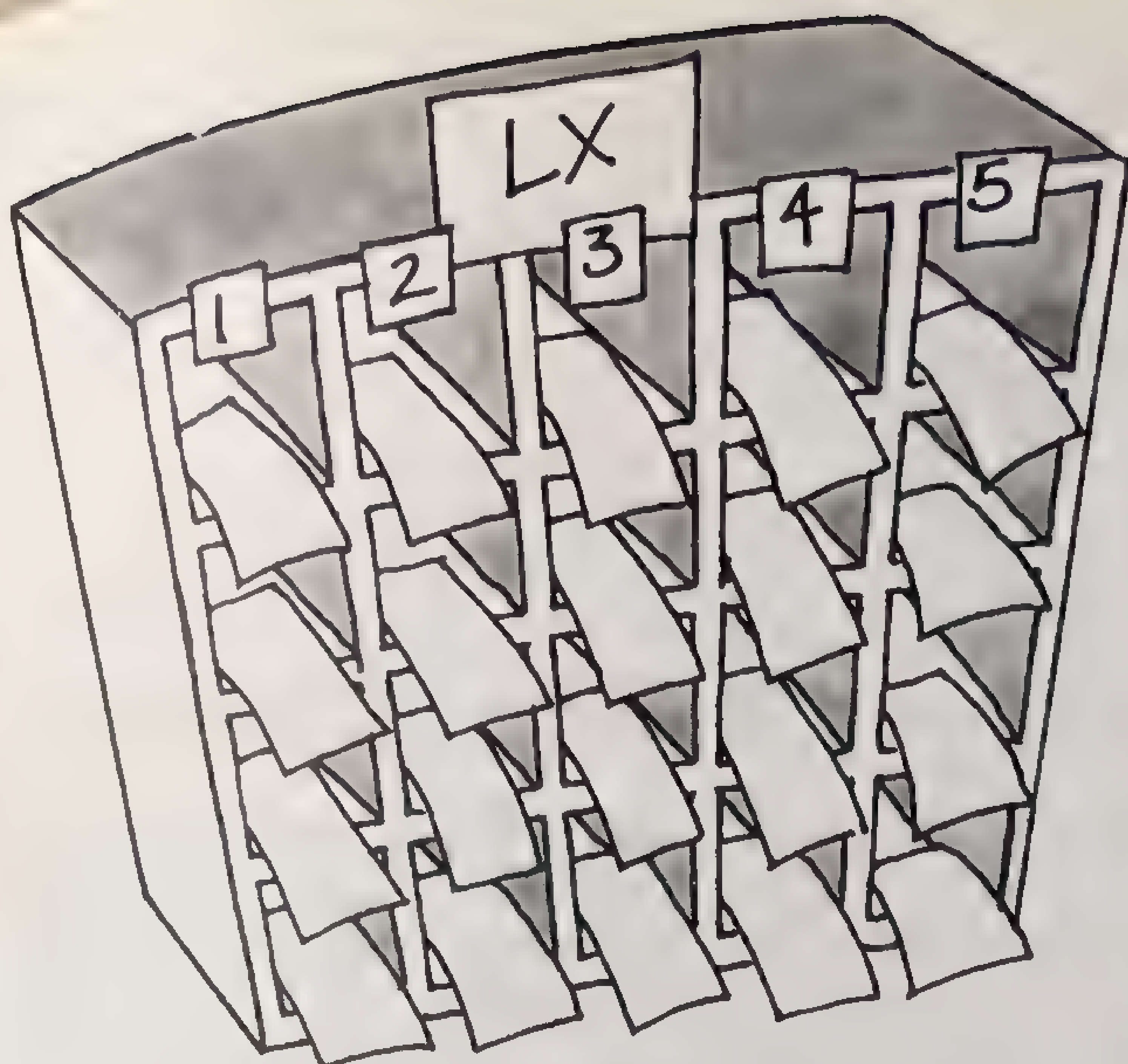


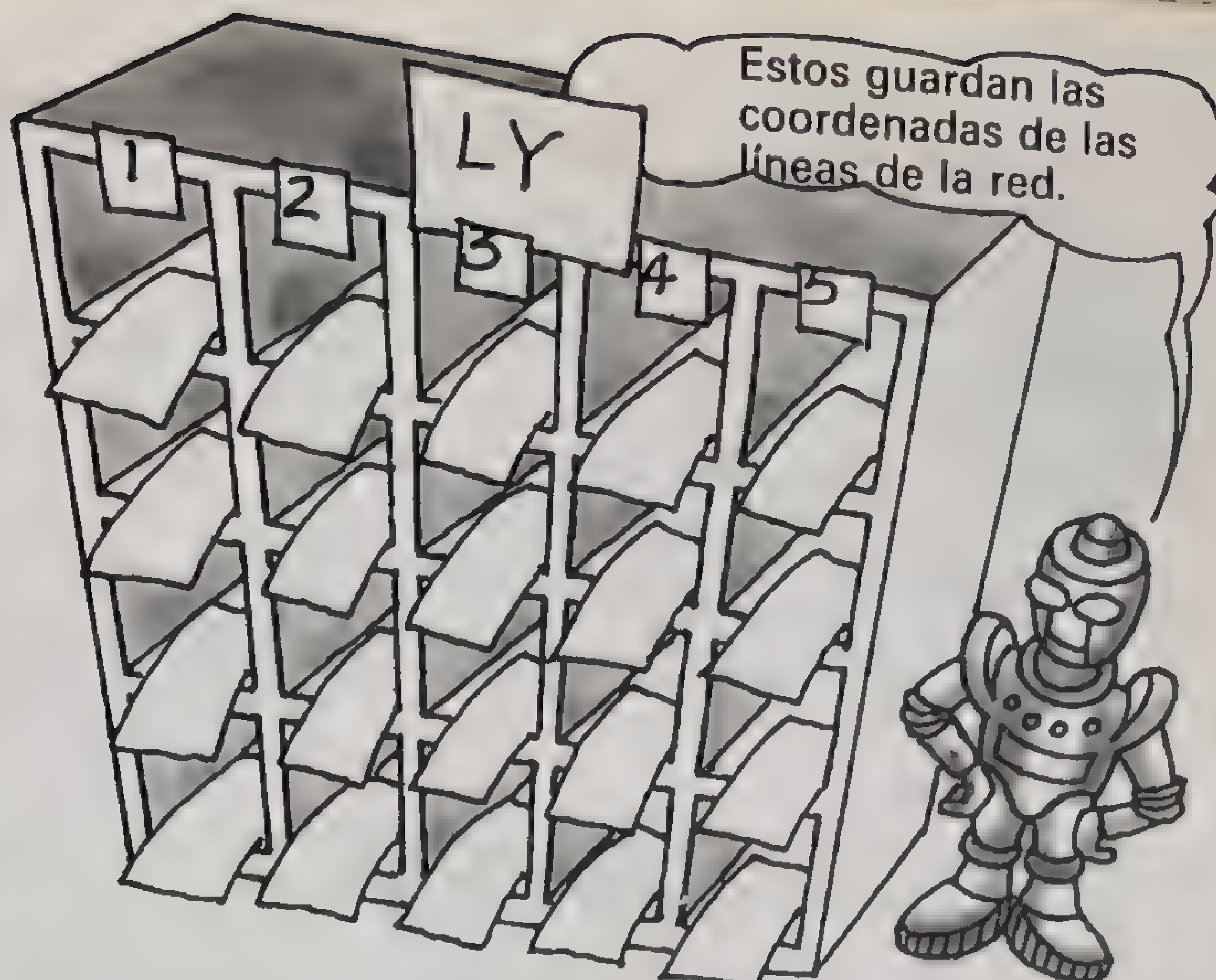
CX guarda las coordenadas X y CY guarda las coordenadas Y.



El programa usa 4 matrices para almacenar todos los datos de las coordenadas. CX y CY guardan las coordenadas X e Y para las cuatro esquinas de la red. Tú le das a la computadora datos para estas matrices al principio del programa CX(1) y CY(1) contienen las coordenadas para la primera esquina de la red, CX(2) y CY(2) para la segunda y así sucesivamente.

* En algunas computadoras como el Spectrum y el ORIC, las coordenadas X e Y para una línea se miden desde el último punto dibujado. Para transformar el programa para estas computadoras ver página 47.





LX y LY son las matrices para almacenar las coordenadas X e Y para las líneas de la red. Son matrices bidimensionales en las que cada matriz tiene cuatro filas. El número de columnas se fija en el programa dependiendo del número de líneas de red que quieras. Cada fila almacena las coordenadas de las líneas de un lado de la red así la fila 1 equivale al lado 1 etc. La computadora guarda los datos en LX y LY a medida que las calcula en el programa.

El programa

```
100 INPUT «¿CUANTAS LINEAS DE RED QUIERES?»; N
```

Prueba 20 para computadoras con gráficas de alta resolución y 5 para las de baja resolución.

```
110 DIM CX(4), CY(4)
115 DIM LX(4, N), LY(4, N)
```

Le dice a la computadora cómo hacer de grandes las matrices.

```
120 CLS
130 REM DIBUJA LOS LADOS DE LA RED
135 REM INSERTA LA INSTRUCCION DEL MODO DE
    GRAFICOS DE TU COMPUTADORA
140 FOR I=1 TO 4
145 READ CX(I), CY(I)
150 NEXT I
```

Los números dados aquí son para la red de la izquierda en la página opuesta, cambia estos números para hacer redes de distintas formas.

Bucle para introducir los datos para las esquinas en CX y CY. Cada vez que se repite el bucle los dos números siguientes en la línea 160 son introducidos en CX y CY.

```
160 DATA 100,50,700,50,500,700,150,500
```

Estas son las coordenadas para las esquinas. Puede que necesites cambiar estos números para que quepan en tu pantalla.

```
170 PLOT CX(4), CY(4)
```

Usa el comando de tu computadora para PLOT y DRAW.

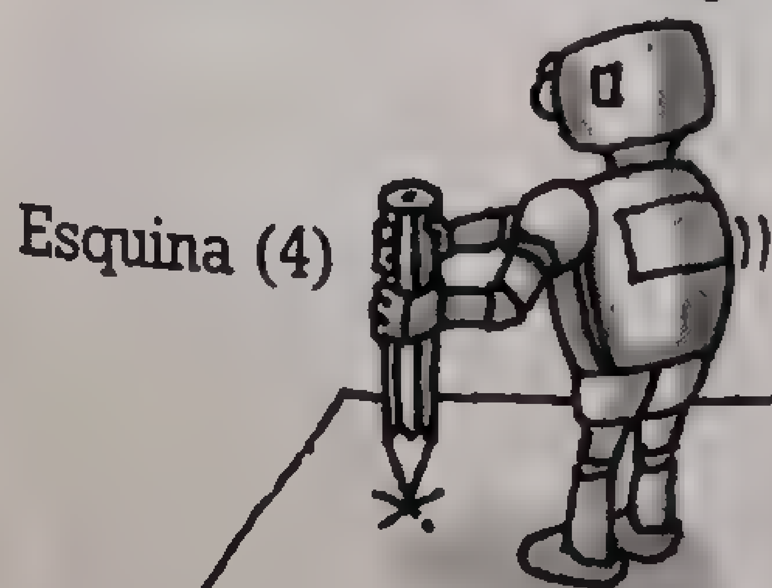
Dibuja la esquina 4 usando los números almacenados en CX(4) y CY(4).

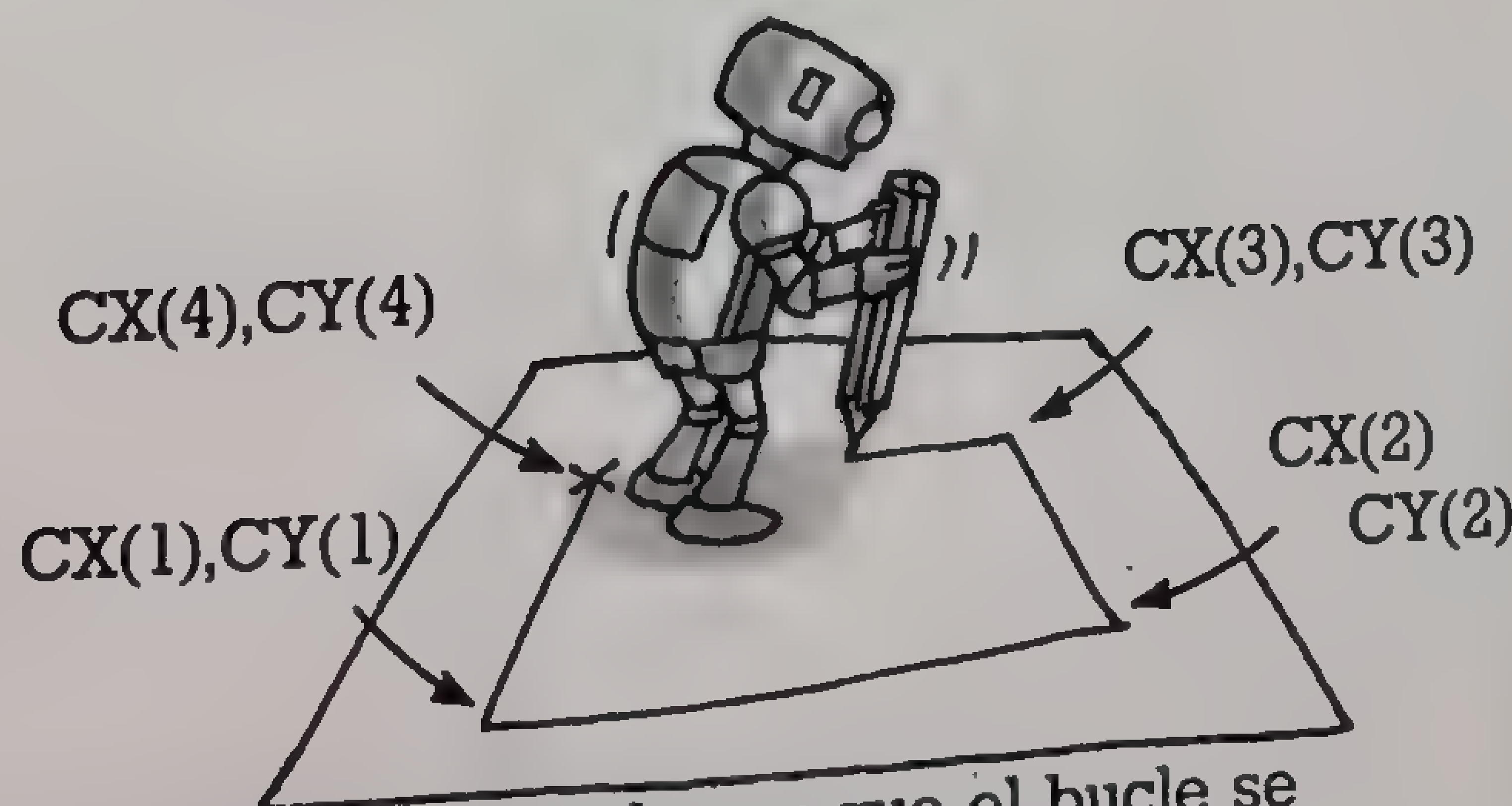
```
180 FOR I=1 TO 4
185 DRAW CX(I), CY(I)
190 NEXT I
```

Bucle para dibujar los lados de la red.

EL LISTADO CONTINUA A LA VUELTA DE LA PAGINA

Dibujando las esquinas

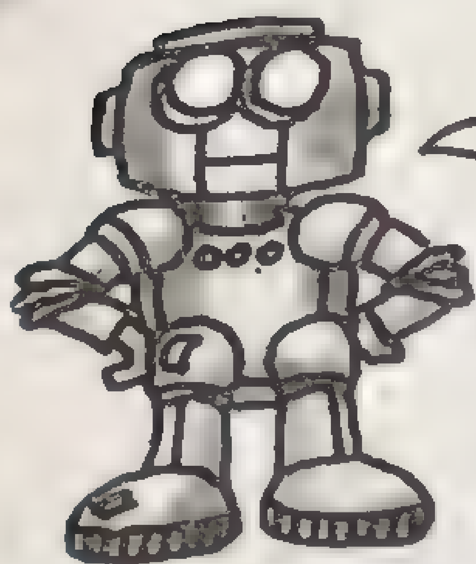




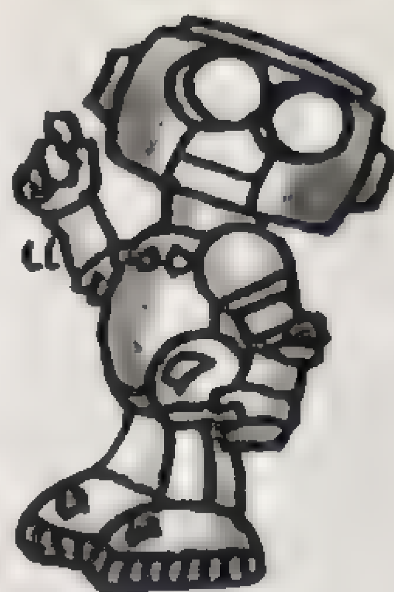
Para dibujar los lados de la red la computadora dibuja primero la esquina 4 (línea 170). Entonces el bucle de las líneas 180-190 hace que dibuje una línea a la esquina 1. Cada vez que el bucle se repite la variable I se incrementa en uno y la computadora dibuja una línea a la esquina siguiente.

```
200 REM CALCULA LAS COORDENADAS PARA LAS LINEAS DE LA RED
```

La parte siguiente del programa en cuatro bucles para calcular las coordenadas de las líneas de la red. Puedes ver como son en los dibujos de abajo.



Recuerda, CX(1) y CX(2) son las coordenadas X para los esquemas 1 y 2, y CY(1) y CY(2) son las coordenadas Y.



```
210 FOR I=1 TO N
```

N es el número de líneas de red que escoges.

```
220 LET LX(1, I)=CX(1)+(CX(2)-CX(1))*I/N
```

Calcula las coordenadas X para las líneas de la red a lo largo del lado 1 y las almacena en LX, fila 1, columnas 1 a N.

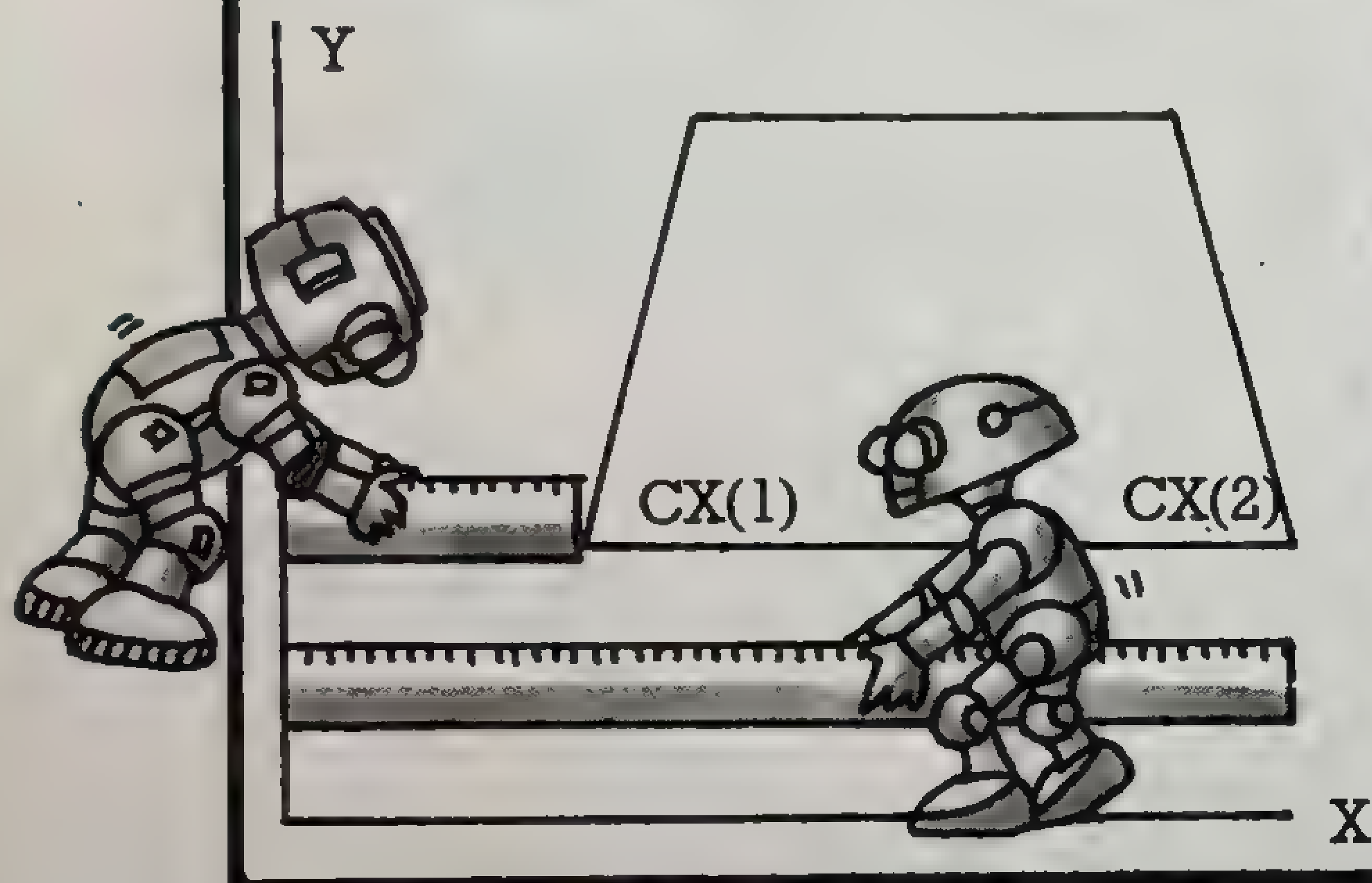
```
230 LET LY(1, I)=CY(1)+(CY(2)-CY(1))*I/N
```

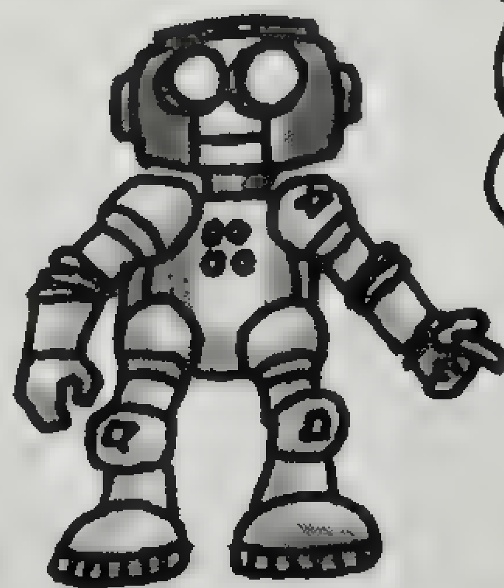
Calcula las coordenadas Y para el lado 1 y las guarda en LY fila 1 columnas 1 a N.

```
240 NEXT I
```

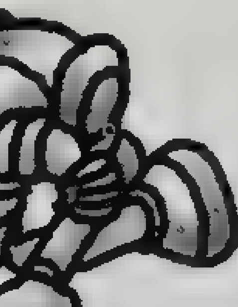
LISTADO CONTINUADO ABAJO

Calculando las coordenadas



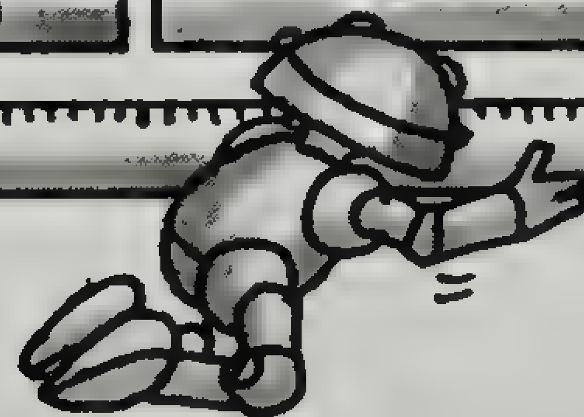


Este es el número de puntos X a lo largo del lado 1.

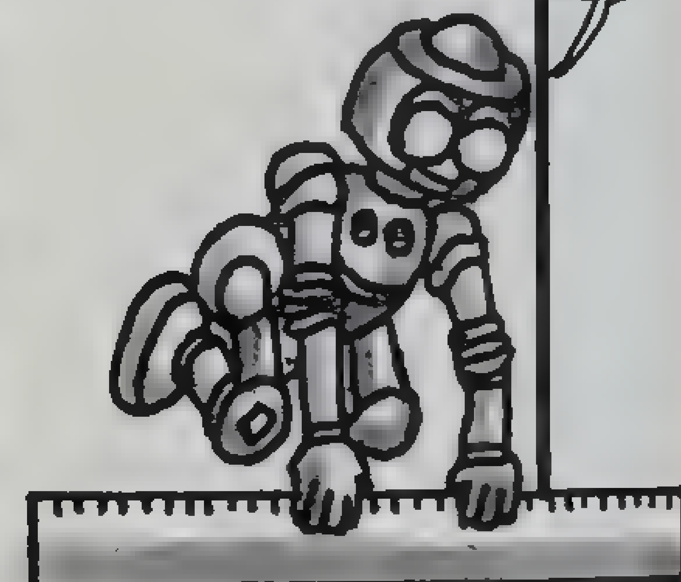


CX(1)

CX(2)



Multiplica por 1/N



Cada bucle calcula las coordenadas para las líneas de la red a lo largo de un lado de la red. Por ejemplo, las líneas 210-240 calculan las coordenadas para el lado 1. En la línea 220 la computadora resta CX(1) de CX(2). Esto le da el número de puntos X a lo largo del lado 1. La primera vez que efectúa el bucle multiplica este número por 1/N (N es el número de líneas de red que escoges). Si N es 5, esto da 1/5 de la longitud del lado 1. Entonces suma este número a CX(1) y almacena el resultado en LX(1,1). La segunda vez que efectúa el bucle I = 2, así que multiplica por 2/5 y almacena la respuesta en LX(1,2). Hace esto para todos los valores de I desde 1 hasta N para encontrar todas las coordenadas X para las líneas de red a lo largo de lado 1. La línea 230 usa el mismo método para encontrar las coordenadas Y.

```
250 FOR I=1 TO N
260 LX(3, I)=CX(4)+(CX(3)CX(4))*I/N
270 LET LY(3, I)=CY(4)+(CY(3)CY(4))*I/N
280 NEXT I
```

Bucle para calcular las coordenadas para el lado 3. Para almacenarlas en el mismo orden como para el lado 1 (esto es de izquierda a derecha) la computadora tiene que hacer las sumas en orden contrario. Resta la esquina 4 de la esquina 3, entonces suma el resultado a la esquina 4.



Prueba a cambiar estas líneas para que se parezca a los otros bucles y mira qué ocurre.

```
290 FOR I=1 TO N
300 LET LX(2, I)=CX(2)+(CX(3)CX(2))*I/N
310 LET LY(2, I)=CY(2)+(CY(3)CY(2))*I/N
320 NEXT I
```

Bucle para calcular las coordenadas para el lado 2.

EL LISTADO CONTINUA EN LA PAGINA OPUESTA

```
330 FOR I=1 TO N
340 LET LX(4, I)=CX(1)+(CX(4)-CX(1))*I/N
350 LET LY(4, I)=CY(1)+(CY(4)-CY(1))*I/N
360 NEXT I
```

Bucle para calcular las coordenadas del lado 4.

```
400 REM DIBUJA LAS LINEAS DE LA RED
410 LET FILA=1: GOSUB 500
420 LET FILA=2: GOSUB 500
```

Esta línea fija una variable llamada fila y le da el valor 1. Entonces la computadora va a la subrutina en la línea 500. Después de la subrutina vuelve a la línea 420 y cambia Fila a 2, entonces vuelve a la subrutina otra vez.

```
430 STOP
```

Esto detiene el programa después de que la subrutina se ha efectuado dos veces.

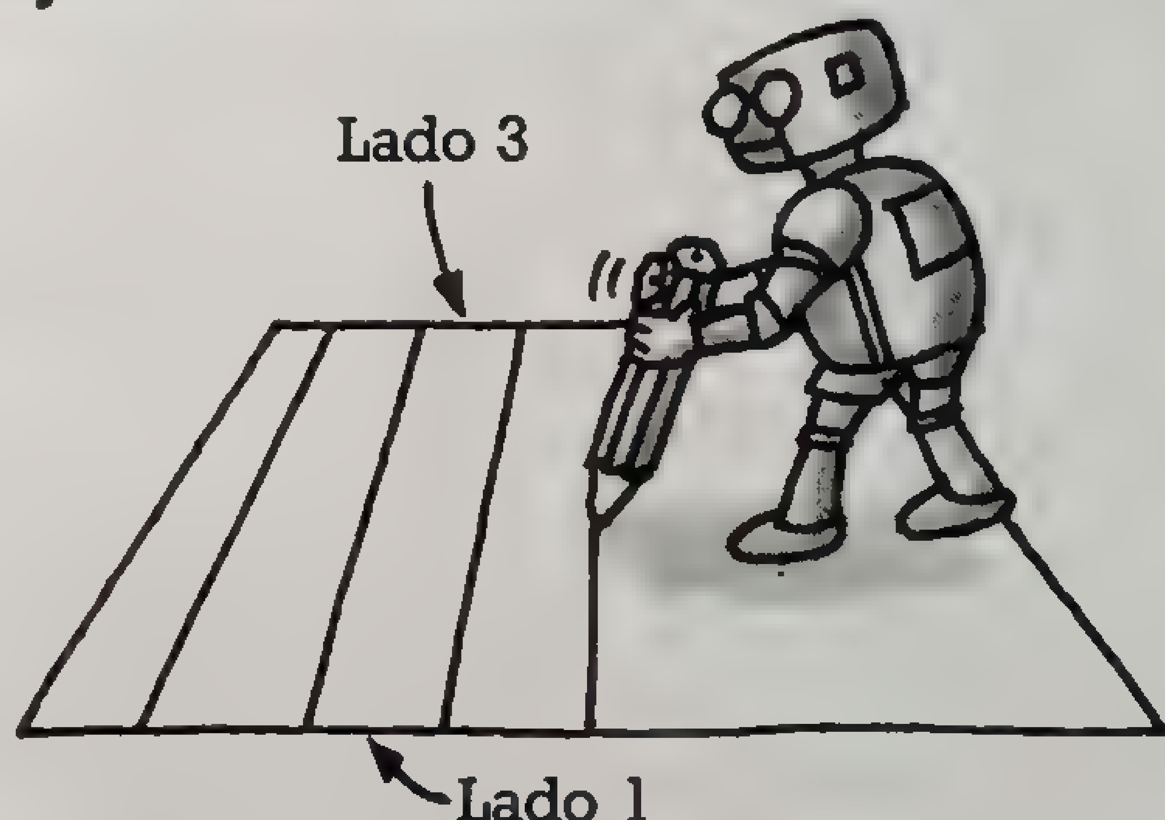
```
500 REM SUBRUTINA
510 FOR I=1 TO N
520 PLOT LX(FILA, I), LY(FILA, I)
530 DRAW LX(FILA+2, I), LY(FILA+2, I)
540 NEXT I
```

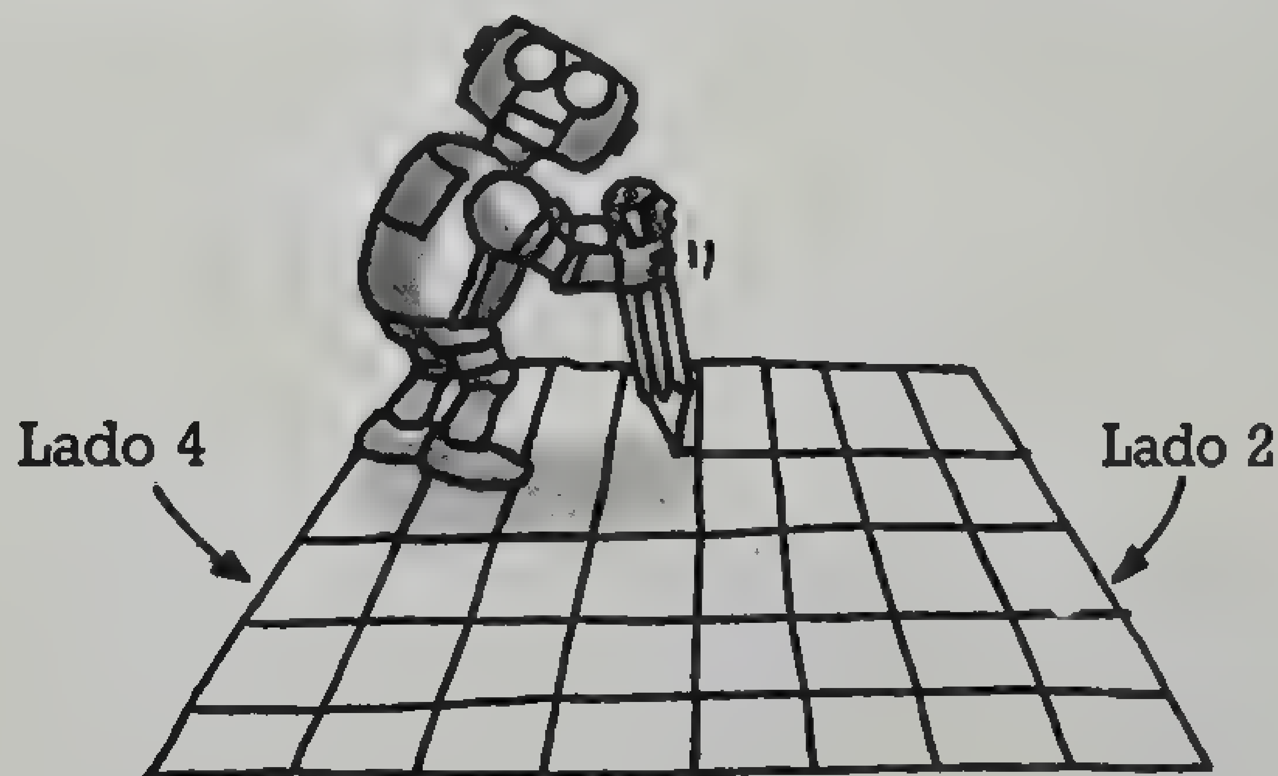
FILA y I son los subíndices para LX y LY. Le dicen a la computadora en qué compartimento buscar para encontrar las coordenadas. X e Y de cada línea. FILA es el número de fila y I la columna. Fila 1 contiene las coordenadas para el lado 1, fila 2 para el lado 2, etc.

```
550 RETURN
```

Vuelta a la línea 420 la primera vez que pasa por la subrutina y la 430 la segunda vez.

Dibujo de las líneas de red





La primera vez que efectúa la subrutina FILA = 1, por lo que en la línea 520 la computadora dibuja un punto en el lado 1. En la línea 530 suma 2 a fila y dibuja una línea al lado 3. La segunda vez que efectúa la subrutina Fila = 2 por lo que dibuja puntos en el lado 2 y dibuja líneas al lado 4.

Ideas para alterar el programa

1. Para hacer redes de diferentes formas, calcula las coordenadas para una forma que te guste primero en un papel. Recuerda que el par de puntos en la línea 160 son las coordenadas para la esquina 1, el segundo par es para la esquina 2, etc. Si haces dos esquinas iguales obtendrás un triángulo. Mira a ver si puedes hacer que los lados se crucen, como se muestra en el dibujo de la derecha en la página 26.

2. Puedes usar INPUT con un bucle para hacer que la computadora te pregunte por los datos. Reemplaza las líneas 140-160 por las líneas siguientes.

```
140 FOR I=1 TO 4
150 PRINT «CUALES SON LAS COORDENADAS PARA LA ESQUINA»; I
155 INPUT CX(I), CY(I)
160 NEXT I
```

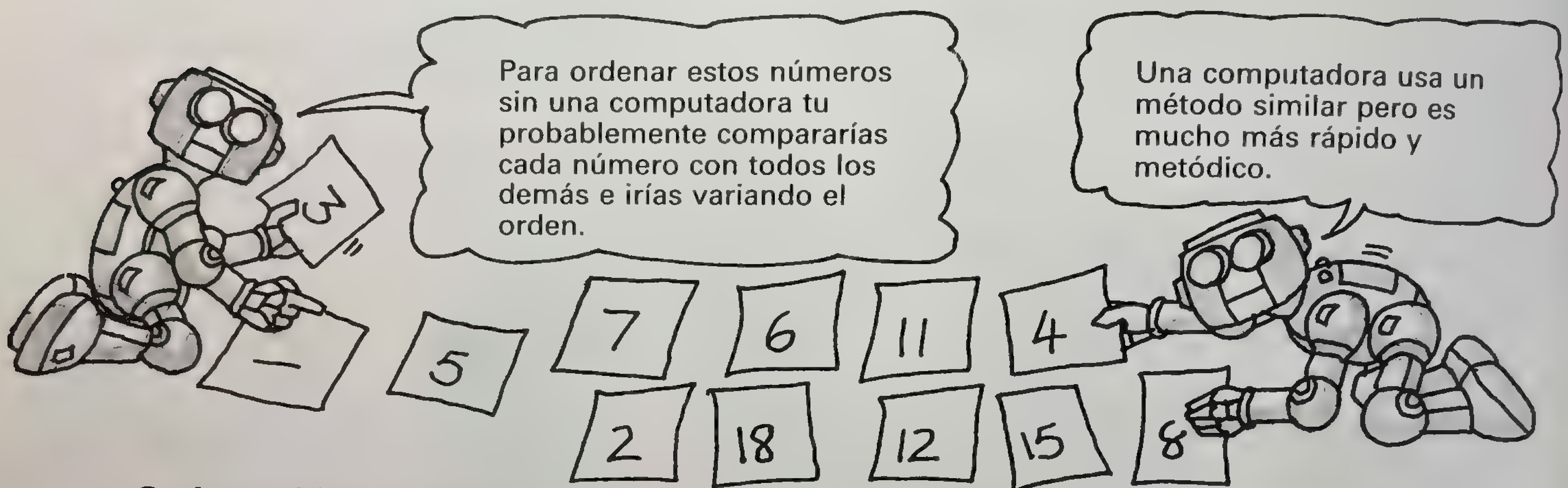
3. Para hacer las líneas de red de colores, introduce el comando de color de tu computadora antes del GOSUB en las líneas 410 y 420. Recuerda poner dos puntos para separar el comando de color del GOSUB.

Programas para ordenar datos

Algunas veces necesitas ordenar datos en orden alfabético o numérico. Por ejemplo, para hacer un índice de una revista, o analizar datos recogidos por ahí sobre el tiempo o vistas o investigaciones sobre la historia local. Las listas cortas son fáciles de ordenar a mano, pero con un gran número de datos una computadora es mucho más rápida y precisa.

Los programas especiales para ordenar datos se llaman «sorts». Hay muchos programas diferentes para ordenar ya escritos un BASIC. Los puedes encontrar en revistas. Los diferentes programas usan técnicas de programación diferentes y se utilizan para trabajos diferentes.

En las páginas siguientes hay dos tipos diferentes de programas de ordenación. Uno se llama una «ordenación de burbuja» (puedes encontrar por qué, abajo) y el otro es una ordenación shell (nombrado así por la persona que lo escribió). Una ordenación de burbujas es de las más lentas y sólo es útil para pequeñas cantidades de datos. Una ordenación shell es mucho más rápida. En la página 35 hay algunas líneas que puedes añadir a los programas para compararlos y ver lo rápida que es tu computadora.



Ordenación de burbuja

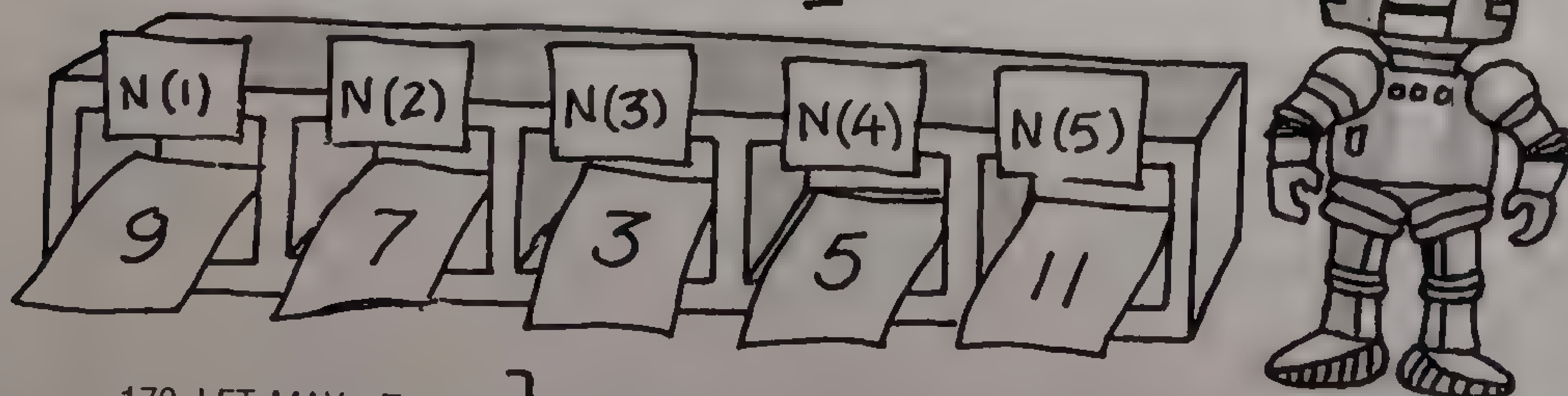
En una ordenación de burbuja la computadora comienza al principio de la lista desordenada y compara los dos primeros elementos. Si están mal ordenados los da la vuelta y compara los dos elementos siguientes. Continúa así por toda la lista de modo que los números mayores van moviéndose gradualmente al final de la lista.

El programa siguiente es una ordenación de burbuja por números. A la vuelta de la página hay otra versión de programa por ordenar palabras.

```
100 REM ORDENACION DE BURBUJA PARA NUMEROS
110 INPUT «¿CUANTOS NUMEROS A ORDENAR?»; T
120 DIM N(T)
130 FOR I=1 TO T
140 PRINT «NUMERO»; I
150 INPUT N(I)
160 NEXT I
```

120: Fija una matriz llamada N con T compartimentos. T es el número total de números que quieres que se ordenen.

130–160: Te pregunta por los números a ordenar y los almacena en la matriz.



En este ejemplo hay 5 números.

```
170 LET MAX=T
```

Fija otra variable llamada Max para llevar cuenta del total ya que T cambiará a lo largo del programa.

```
175 LET X=0
```

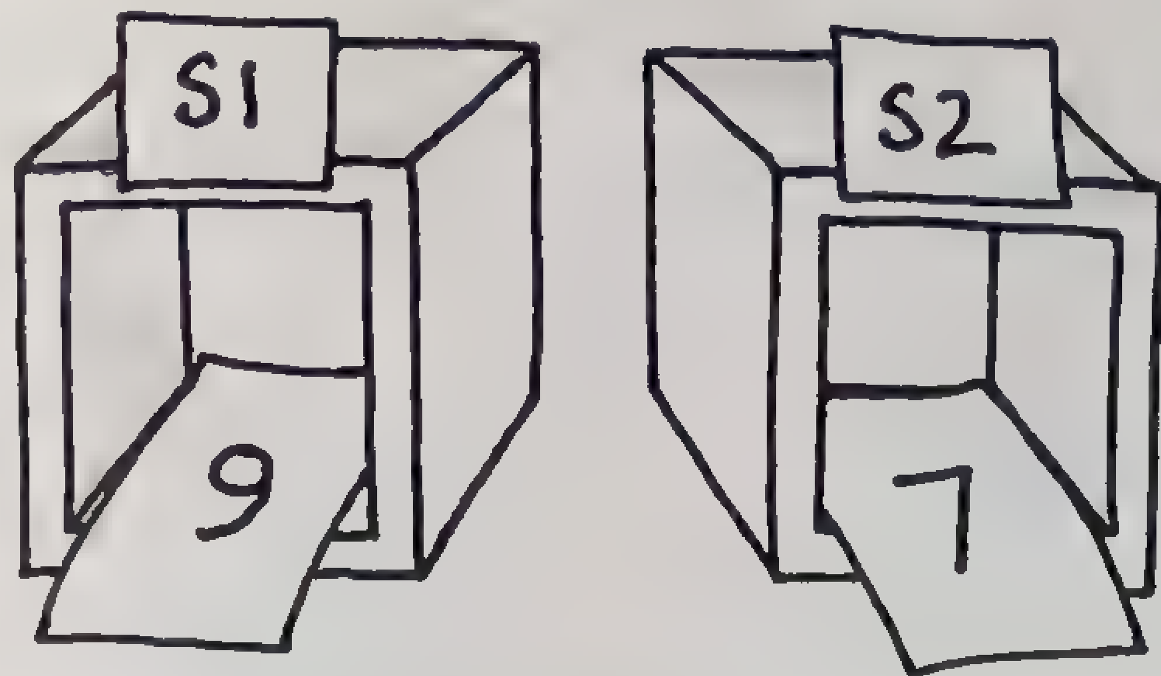
X es un contador.

```
180 FOR C=1 TO T-1
```

Bucle para efectuar T-1 veces. Este es el número de veces requerido para comparar cada pareja en la lista.

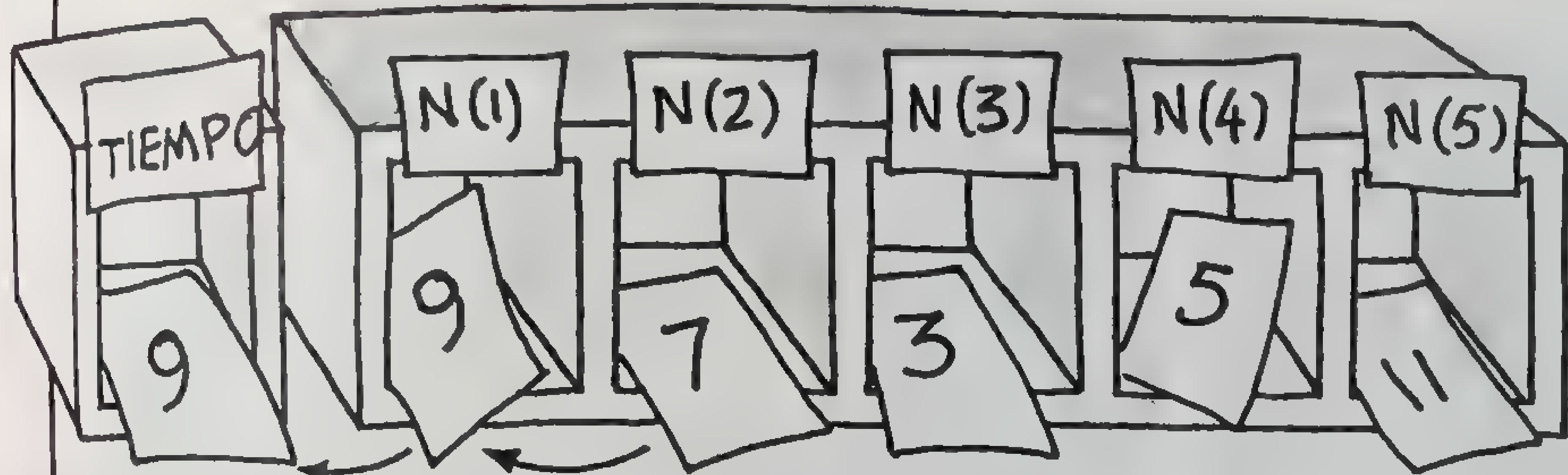
```
190 LET S1=N(C): LET S2=N(C+1)
```

Las variables S1 y S2 son para almacenar cada par de números mientras son comparados. La primera vez que se efectúa el bucle C = 1 así que N1 y N2 se introducen en S1 y S2.




```
200 IF S1<=S2 THEN GOTO 250
```

Compara los dos números. Si S1 es menor que S2 están en el orden correcto así que la línea 250 manda a la computadora al principio del bucle a seleccionar el siguiente par de números. Si S1 es mayor que S2 la computadora continúa con las líneas siguientes que cambiaran la posición de estos números en la matriz.




```
210 LET TEMP=N(C)
```

El número N(C) es introducido en una variable llamada TEMP.

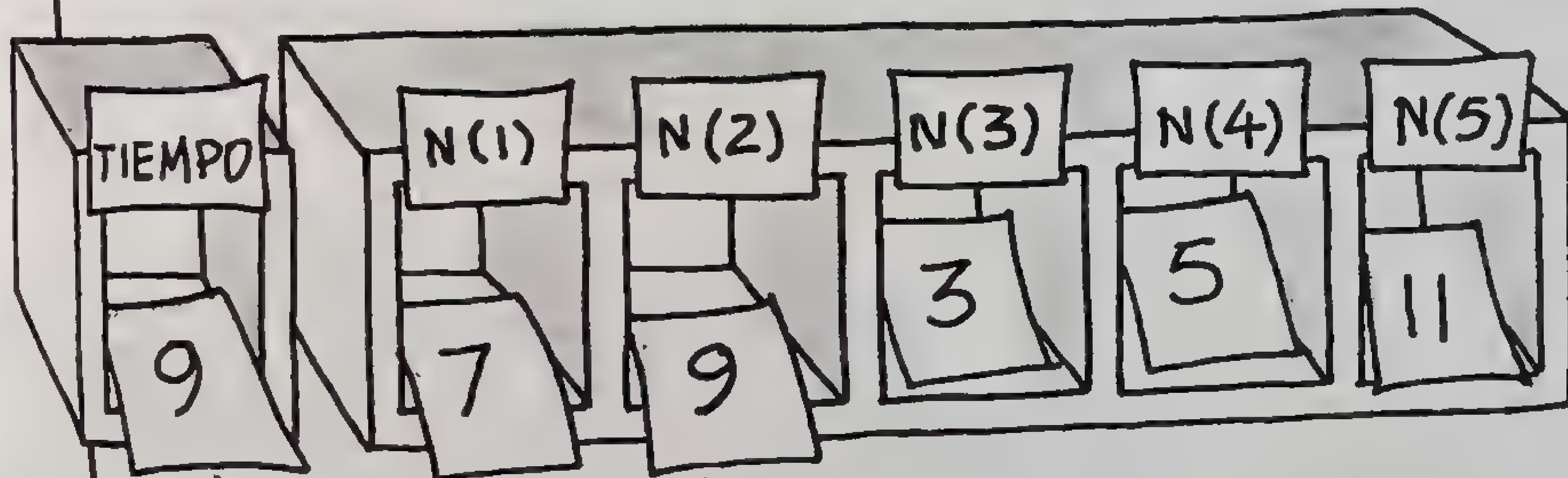
```
220 LET N(C)=N(C+1)
```

El número en N(C + 1) es movido a la posición N(C) en la matriz.

```
230 LET N(C+1)=TEMP
```

El número en TEMP es puesto en la posición N(C + 1).

Cada vez que el bucle se repite el número se moverá un lugar a la derecha hasta que esté en su posición correcta.



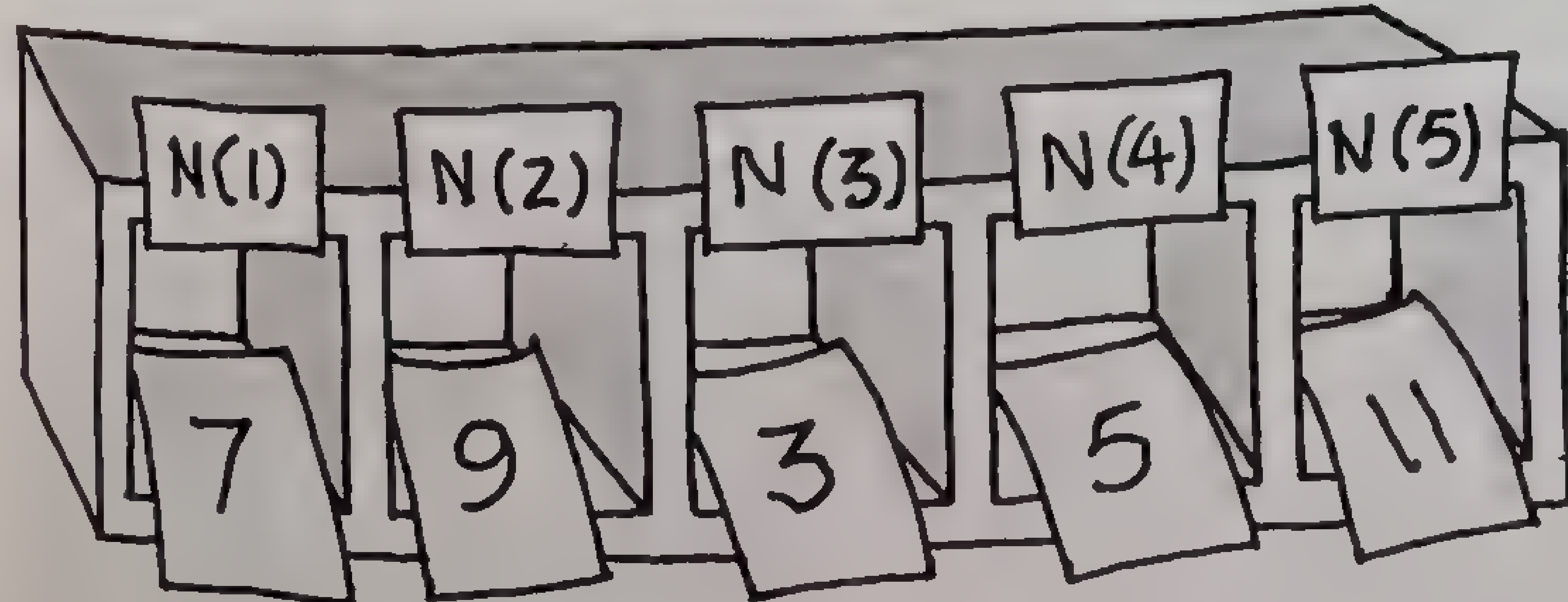

```
240 LET X=X+1
```

Añade 1 a X para indicar que se ha producido un cambio.

```
250 NEXT C
```

Manda a la computadora de vuelta a comparar el siguiente par de números. Cuando ha repetido el bucle T − 1 veces la computadora ha comparado todos los números S una vez y continúa con la línea 260.

La segunda vez que se efectúa el bucle C = 2, así que N(C) = N(2) y N(C + 1) = N(3).




```
260 IF X>0 THEN LET T=T-1: GOTO 175
```

Si X es más de 0 se ha efectuado un cambio así que la computadora resta 1 de T ya que ya hay un número en su posición correcta. Entonces vuelve al principio del bucle. Si X = 0 los números están en su posición correcta y la computadora va a la línea 270.

```
270 PRINT LOS NUMEROS ORDENADOS SON
280 FOR I=1 TO MAX
290 PRINT N(I)
300 NEXT I
```

MAX es el número total de números ordenados.

Repite el IF...THEN si pones el GOTO de la línea 260 en una nueva línea.

Ordenación de burbuja para palabras

El programa siguiente es una ordenación de burbuja para palabras. Es igual que la ordenación de burbuja de números excepto que las variables para almacenar los datos (N, S1, S2 y TEMP) son variables de cadena.*

La computadora usa el mismo método para comparar letras como para números. Dentro de la computadora las letras y símbolos están representados por números, así que cuando le das a la computadora caracteres a comparar, compara sus números de código. Para comparar dos palabras, comprueba la primera letra de cada una y si son iguales compara la segunda y después la tercera y así sucesivamente. Puedes introducir números así como letras en las variables de cadena, así que puedes usar la ordenación de burbuja para información que contenga palabras y números como direcciones.

```
¿CUANTOS ELEMENTOS A ORDENAR? 4
ELEMENTO 1
UNIDADES DE DISCO 34 76 82 93
ELEMENTO 2
CODIGO DE MAQUINA 55 72 85
ELEMENTO 3
SONIDO 32
ELEMENTO 4
GRAFICOS 8 23 45
LA LISTA ORDENADA ES
UNIDADES DE DISCO 34 76 82 93
GRAFICAS 8 23 45
CODIGO DE MAQUINA 55 72 85
SONIDO 32
```

```
¿CUANTOS ELEMENTOS A ORDENAR? 4
ELEMENTO 1
KELLER MARY 12 PANSY PLACE
ELEMENTO 2
SMITH JOHN QUEEN STREET
ELEMENTO 3
JONES PETER 3356 WESTSIDE
ELEMENTO 4
FLAK KANE 34 RING ROAD
LA LISTA ORDENADA ES
FLAK JANE 34 RING ROAD
JONES PETER 3356 WESTSIDE
KELLER MARY R. PANSY PLACE
SMITH JOHN 12 QUEEN STREET
```

En este ejemplo la computadora está almacenando elementos para un índice, y a la derecha direcciones. Los elementos fueron tecleados sin comas ya que para la mayoría de las computadoras, una coma es un separador o delimitador entre los diferentes elementos de información. Si quieres usar comas en cadenas, pon la cadena entre comillas.

El programa

```
100 REM ORDENACION DE BURBUJA PARA PALABRAS
110 INPUT «¿CUANTOS ELEMENTOS A ORDENAR?»; T
120 DIM N$(T)
130 FOR I=1 TO T
140 PRINT «ELEMENTO»; I
150 INPUT N$(I)
160 NEXT I
170 LET MAX=T
175 LET X=0
180 FOR C=1 TO T-1
190 LET S1$=N$(C): LET S2$=N$(C+1)
200 IF S1$<=S2$ THEN GOTO 250
210 LET TEMP$=N$(C)
220 LET N$(C)=N$(C+1)
230 LET N$(C+1)=TEMP$
240 LET X=X+1
250 NEXT C
260 IF X>0 THEN LET T=T-1: GOTO 175
270 PRINT «LA LISTA ORDENADA ES»
280 FOR I=1 TO MAX
290 PRINT N$(I)
300 NEXT I
```

Fija una matriz llamada N$ con T compartimentos. (línea 120)

Prueba a introducir elementos que empiecen con un símbolo, un número, una letra mayúscula y una letra minúscula y mira el orden en que tu computadora los ordena.

Los dos primeros elementos se introducen en S1$ y S2$. (línea 190)

Compara S1$ y S2$. (línea 200)

Cambia las posiciones de los dos primeros elementos N$. (líneas 210–230)

Si X > 0 resta 1 del total y vuelve al principio del bucle para comprobar la lista otra vez. (línea 260)

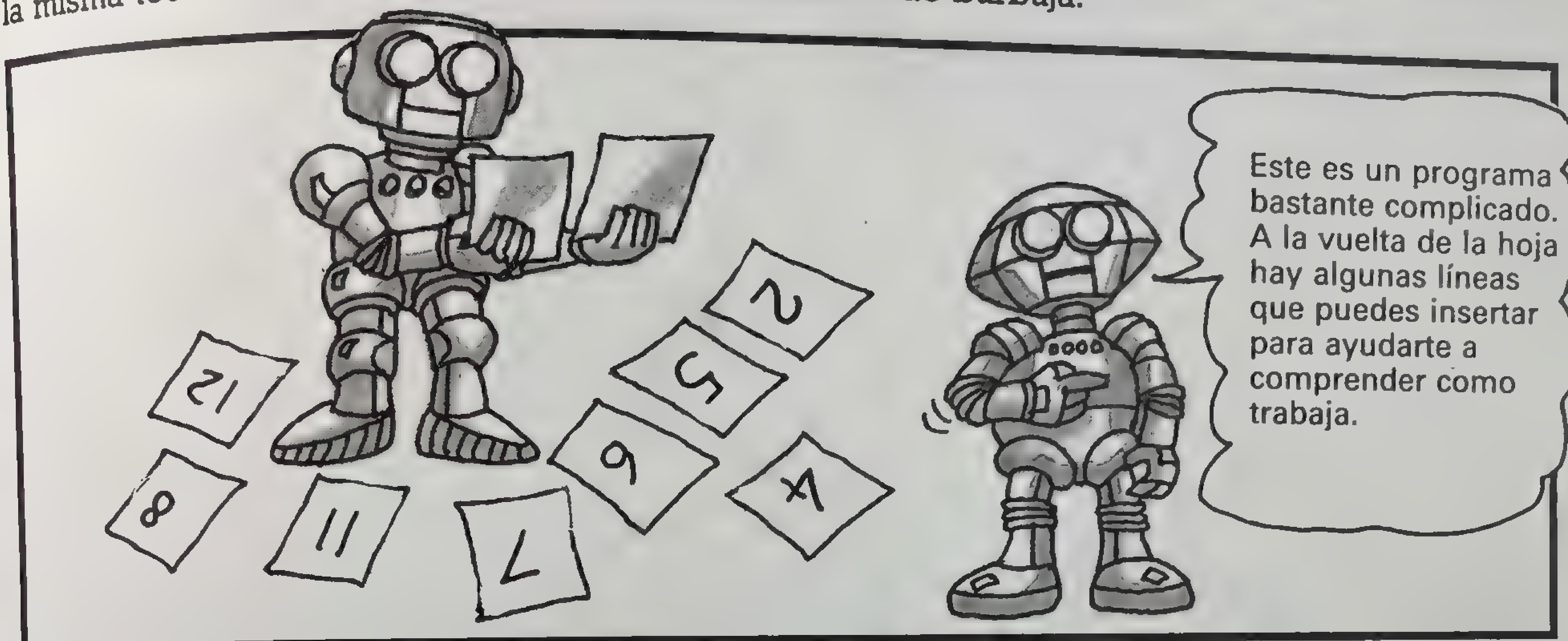
Repite el IF...THEN si pones el GOTO de la línea 260 en una nueva línea.

* Para computadoras Sinclair (Timex) cambia la línea 120 a DIM N$ (T, N) donde N es la longitud de la cadena más larga que quieras introducir.

Ordenación Shell

Si quieres ordenar muchos elementos, la ordenación de burbuja es muy lenta. Puede invertir casi un minuto para ordenar cincuenta elementos en algunas computadoras. El programa siguiente es una ordenación Shell para números. Es prácticamente tres veces más rápida que la ordenación de burbuja.

En una ordenación Shell la computadora divide la lista de números a ordenar en dos y compara todos los elementos de una mitad con los de la otra mitad. Entonces divide la lista en dos y hace unas comparaciones moviendo los números mayores hacia arriba en la lista usando la misma técnica de movimiento que en la ordenación de burbuja.



El programa

```
100 REM ORDENACION SHELL PARA NUMEROS
110 INPUT «¿CUANTOS NUMEROS A
    ORDENAR?»; T
120 DIM N(T)
130 FOR I=1 TO T
140 PRINT «NUMERO»; I
150 INPUT N(I)
160 NEXT I
```

Estas líneas son como en la ordenación de burbuja. T es el número total de números a ordenar y las líneas 120 a 160 te preguntan por los números y los almacenan en la matriz N.

```
170 LET C=T
```

Fija la variable C igual al número total de elementos.

```
180 LET C=INT(C/2)
```

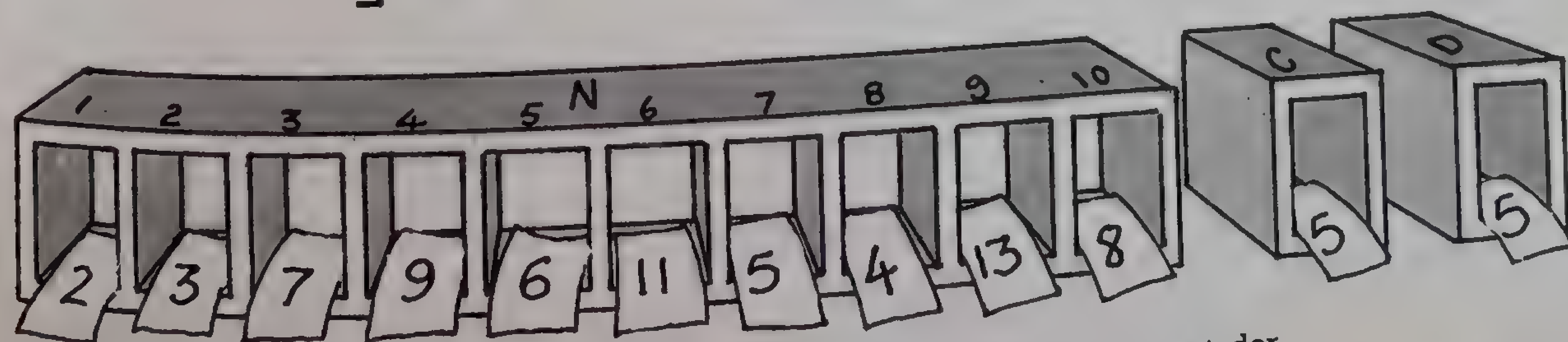
Divide C en dos para dar el número de elementos en la primera mitad de la lista. INT hace que la computadora descarte cualquier número detrás de la coma decimal para hacer de C un número entero.

```
190 IF C=0 THEN GOTO 330
```

El programa divide repetidamente C en dos y cuando C = 0, la computadora va a la línea 330 a imprimir la lista ordenada.

```
200 LET D=T−C
```

D es el número de elementos en la segunda parte de la lista.




```
210 LET E=1
```

E es un contador.

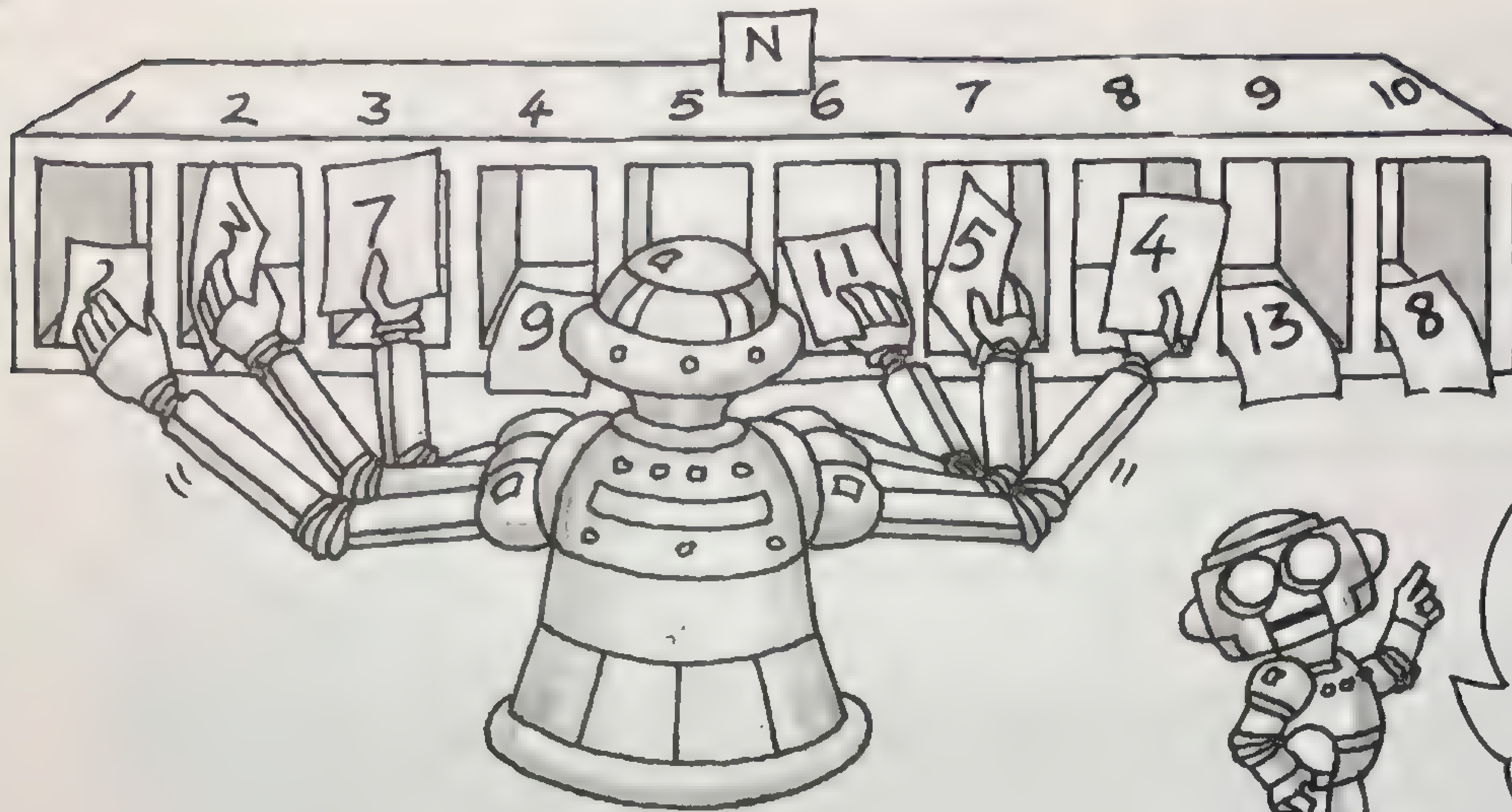
```
220 LET F=E
230 LET G=F+C
```

Las variables F y G son para fijar los subíndices de N a la matriz donde se guardan todos los números.

EL LISTADO CONTINUA A LA VUELTA DE LA PAGINA

```
240 IF N(F)<=N(G) THEN GOTO 300
```

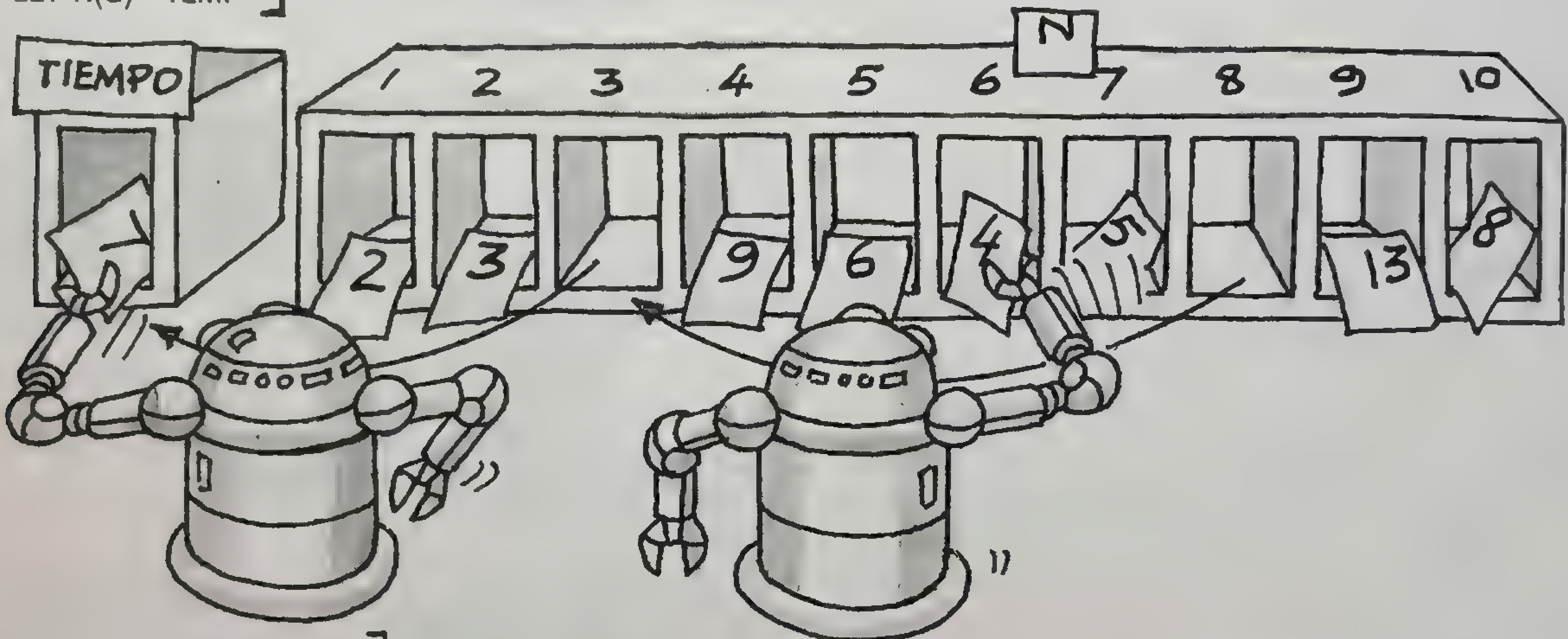
Si el número en N(F) es menor que el número en N(G) la computadora va a la línea 300, suma uno a E y vuelve para reponer F y G para los subíndices del siguiente par de números.



En este ejemplo hay diez números a ordenar. La primera vez que se realiza el programa C = 5 la computadora compara los números en los subíndices 1 a 5 con los de 6 a 10.

```
250 LET TEMP=N(F)
260 LET N(F)=N(G)
270 LET N(G)=TEMP
```

Si N(F) es mayor que N(G) la computadora los cambia.




```
280 LET F=F-C
290 IF F>0 THEN GOTO 230
300 LET E=E+1
```

Comprueba el valor de F entonces añade uno al contador E para cambiar los valores de F y G en las líneas 220 y 230.

```
310 IF E>D THEN GOTO 180
```

Se asegura de que E está dentro de l parte D de la lista. Si no es así, vuelve a la línea 180 a dividir la lista otra vez.

```
320 GOTO 220
```

Si E es menor que D, la computadora va a la línea 220 a fijar los subíndices para el siguiente par de números.

```
330 PRINT «LA LISTA ORDENADA ES»
340 FOR I=1 TO T
350 PRINT N(I)
360 NEXT I
```

Imprime la lista ordenada.

¿Puedes insertar nuevas líneas para que la computadora, una vez haya ordenado una lista te diga cuántas comparaciones y cambios fueron hechos? (Respuesta página 48). Entonces podrías probar el programa con listas diferentes y ver cómo varían.

Para transformar este programa para que ordene cadenas, cambia la variable N por N$ (en las líneas 120, 150, 240-270 y 350), cambia TEMP por TEMP$ (líneas 250 y 270) y reescribe las sentencias PRINT. Si tienes una computadora Sinclair, cambia la línea 120 por DIM N$ (T, N) donde N es la longitud de la cadena más larga que deseas introducir.

¿Cómo funciona?

Para tener una idea más completa de cómo funciona el Shell sort puedes añadir estas líneas adicionales. Imprimen los valores de las variables así que puedes ver qué números está comparando la computadora.

```
233 PRINT
235 PRINT «F=»; F; «, G=»; G
237 PRINT «COMPARAR N(«; F; ») y N(«; G; »)»
245 PRINT «CAMBIAR N(«; F; ») y N(«; G; »)»
274 PRINT «LISTA=»;
275 FOR J=1 TO T
276 PRINT N(J); «   »;
277 NEXT J
278 PRINT
279 INPUT «PRESIONA RETURN PARA CONTINUAR»; Z$
```

F y G son los subíndices de los números a comparar.

Te dice los subíndices de los números a cambiar.

Imprime el orden actual de la lista.

Comparación de los sistemas de ordenación

Si comparaste las ordenaciones de burbuja y shell con sólo unos pocos números, puede que no hayas notado lo mucho más rápido que es el Shell sort. Para comparar los dos sistemas de ordenación podrías hacer que los dos generaran una lista de números aleatorios, y entonces, contar cuánto tiempo invierte cada programa en ordenarlos. A mayor lista de números, la diferencia de tiempo entre los dos sistemas de ordenación crece. A la vuelta de la página hay algunos programas para dibujar gráficos que muestren las diferencias entre los dos sistemas de ordenación.

Generando los números

```
140 LET N(I)=INT (RND(1)*200+1)
150 PRINT N(I)
165 INPUT «PREPARA TU RELOJ Y PRESIONA RETURN
    PARA COMENZAR LA ORDENACION»; Z$
```



Podrías cambiar este número por cualquier otro que quieras.

Para hacer que los programas generen su propia lista de números a ordenar necesitas reemplazar las líneas 140 y 150 en ambos programas e insertar una nueva línea 165 para que puedas controlar cuando empieza la ordenación.

La línea 140 genera números aleatorios entre 1 y 200 y los almacena en la matriz N. La línea 150 imprime los números en la pantalla y la línea 165 hace que el programa espere hasta que presiones RETURN.

Test de ejecución

Para comprobar cada programa debes ejecutarlo varias veces. La primera vez para ver cuánto tarda en ordenar, digamos, 10 números, después 20, 30, 40, etc. Las diferentes marcas de computadora ordenará las listas a diferentes velocidades y algunas, como el ZX81 tienen un modo rápido y lento. Las siguientes son las velocidades en Apple II.

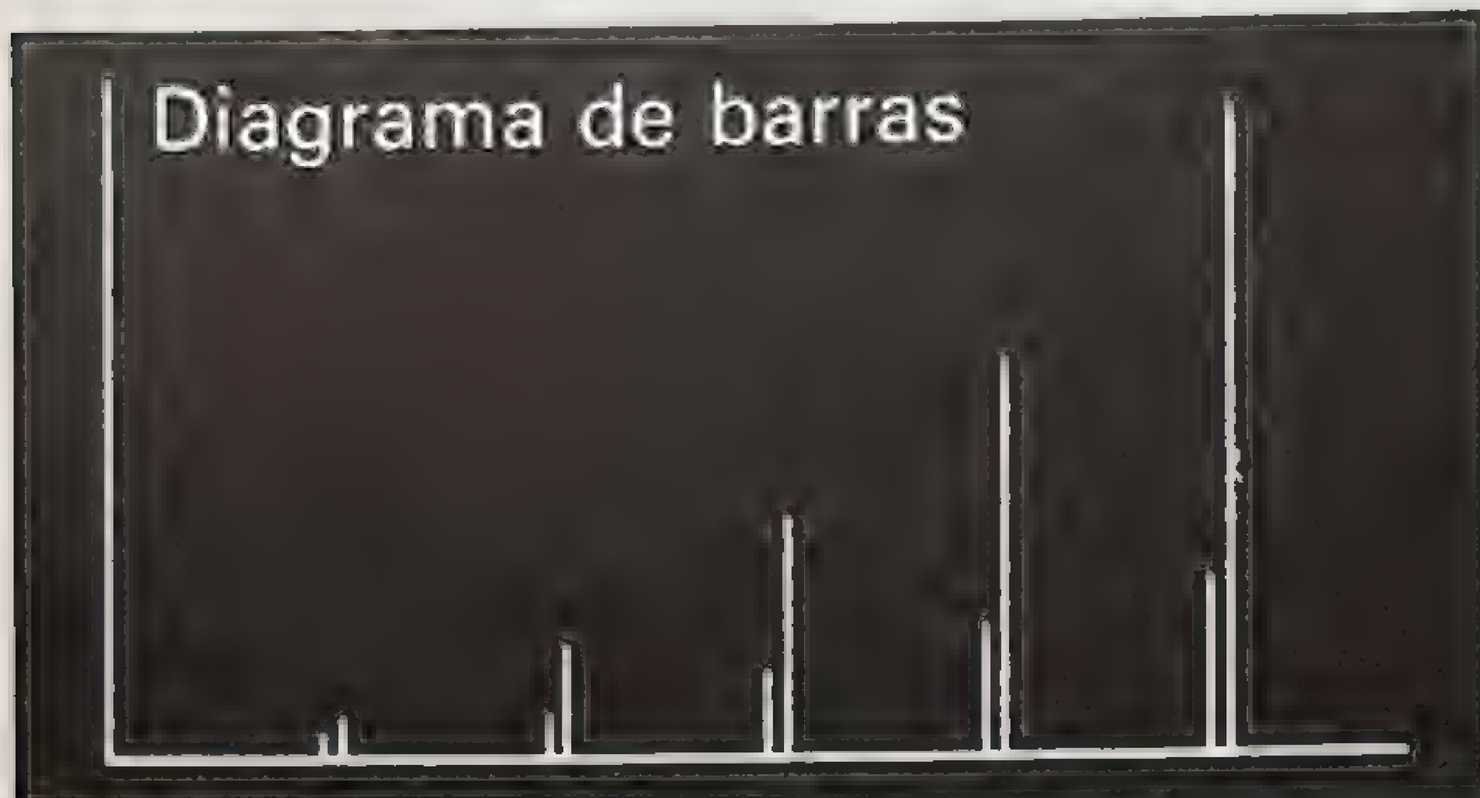
Test de sistemas de ordenación

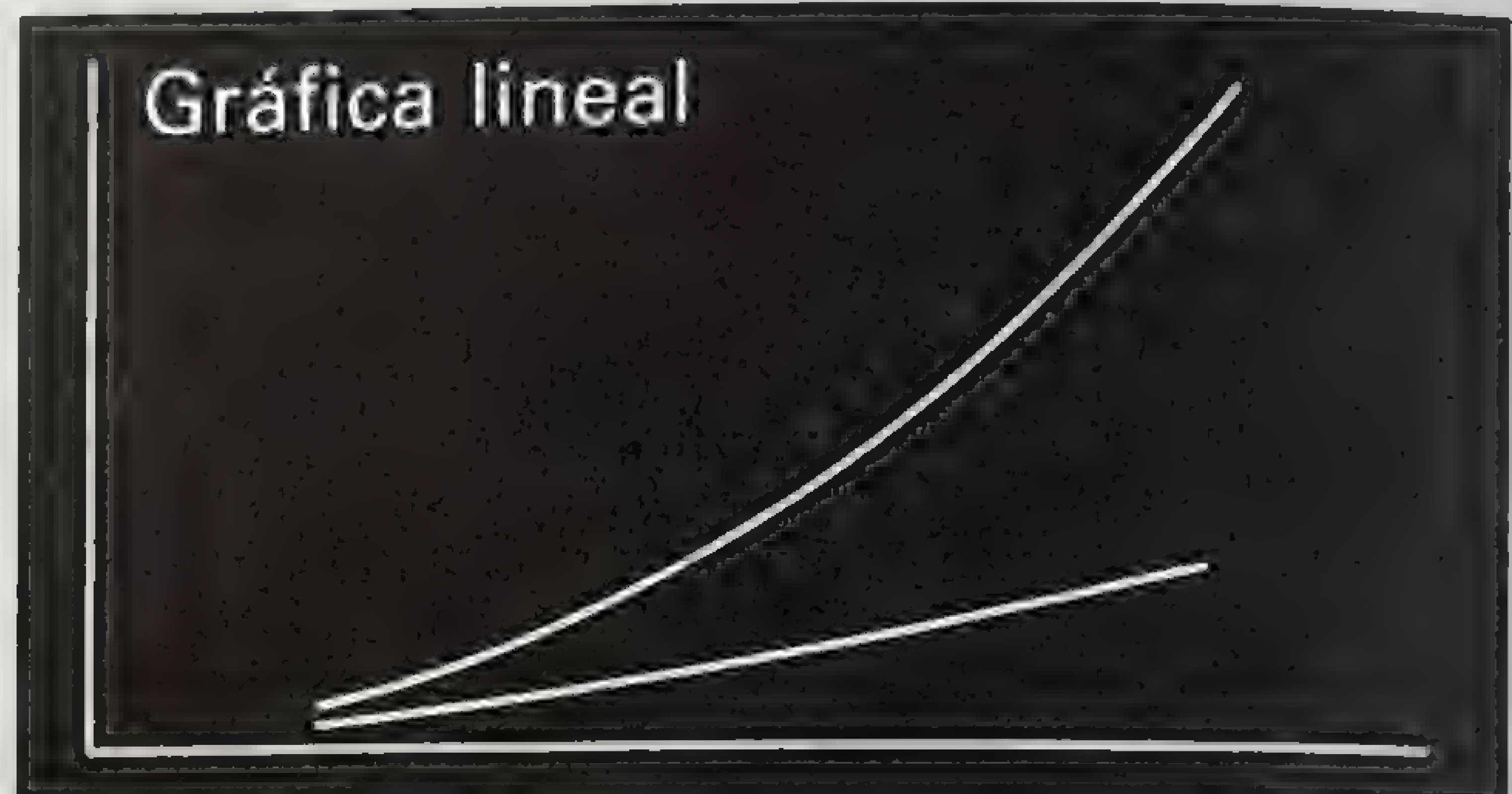
N.º de números ordenados	10	20	30	40	50
Ordenación de burbuja	2 seg	5 seg	11 seg	18 seg	29 seg
Shell	1 seg	2 seg	4 seg	6 seg	8 seg

Dibujo de gráficos

Los resultados de una computadora son mucho más fáciles de leer y comprender si los presentas de una manera interesante, usando tanto gráficos como palabras. Abajo hay un programa de un diagrama de barras para mostrar la diferencia entre las ordenaciones de burbuja y shell. En la página opuesta puedes encontrar cómo transformar el programa de diagrama de barras para que dibuje una gráfica lineal.

Los programas son bastante flexibles y podrías adaptarlos fácilmente para representar informaciones diferentes. También los podrías mejorar añadiendo el comando de color de tu computadora para que dibujara las gráficas a diferentes colores.





Estas son las exposiciones en pantalla para los dos programas de gráficos. Ambos gráficos comparan los tiempos invertidos por los dos sistemas de ordenación para ordenar 10, 20, 30, 40 y 50 números. El tiempo se representa en el eje de la Y y el número de números ordenados está a lo largo del eje X. Si tu computadora puede imprimir texto y posiciones pixel puedes añadir indicaciones para hacer los gráficos más claros. Puedes encontrar cómo hacer esto para el BBC en la página opuesta.

Programa de Diagrama de Barras

```
100 DIM B(5): DIM S(5)
110 LET N=0
120 FOR I=10 TO 50 STEP 10
130 LET N=N+1
140 PRINT PARA «; I; «NUMEROS»
150 INPUT «¿CUANTOS SEGUNDOS INVIRTIO LA ORDENACION DE BURBUJA?»; B(N)
160 INPUT «¿Y LA ORDENACION SHELL?»; S(N)
170 NEXT I
180 INPUT «¿CUANTAS POSICIONES DE DIBUJO HACIA ARRIBA DE TU PANTALLA?»; H
190 INPUT «¿Y CUANTOS A LO LARGO?»; W
200 REM DA EL COMANDO DE MODO DE GRAFICOS SI ES NECESARIO
```

Fija las matrices B y S para almacenar los datos para las ordenaciones de Burbuja y Shell.

Bucle para introducir datos en las matrices. N es un contador para fijar los subíndices para las matrices.

Nota como la variable del bucle I también se usa para contar los números.

Los números para PLOT y DRAW se miden desde el borde de la pantalla. Si tu computadora necesita coordenadas que miden desde el último punto dibujado, ver página 47 para como cambiar el programa.

```
210 REM DIBUJA GRAFICOS
220 PLOT 1, H: DIBUJA 1, 1: DIBUJA W, 1
230 REM DIBUJA GRAFICOS
240 LET X=(W*0.75)/5
250 LET Y=(H*0.75)/B(5)
```

Dibuja los ejes a un pixel del borde de la pantalla. H y W son la altura y anchura de la pantalla.

Los números para X e Y fijan la escala para el gráfico a lo largo de los ejes X e Y. X es tres cuartas del ancho dividiendo por 5 veces el número de mediciones para cada sistema de ordenación. Y es tres cuartas de la altura dividido por el tiempo más largo, esto es B(5).

EL LISTADO CONTINUA EN LA PAGINA OPUESTA

```
260 FOR I=1 TO 5
```

Inicia un bucle para dibujar las barras. La variable del bucle I cuenta las mediciones y cada vez que el bucle se repite la computadora dibuja una barra para cada sistema de ordenación.




```
270 PLOT INT (I*X), 1
```

Dibuja un punto I * X a lo largo y 1 pixel hacia arriba. La primera vez que se efectúa el bucle I y es 1 y X es valor para la escala a lo largo del eje X.

```
280 DRAW INT (I*X), INT (B(I)*Y)
```

Dibuja una línea a B(I) * Y. La primera vez que se efectúa el bucle I es 1 así que B(I) es el tiempo invertido para 10 números e Y es el valor para la escala a lo largo del eje Y.

```
290 PLOT INT (I*X-4), 1
```

Dibuja un punto I * X − 4 puntos a lo largo, esto es, 4 puntos a la izquierda de la barra es la ordenación de burbuja.




```
300 DRAW INT (I*X-4), INT (S(I)*Y)
```

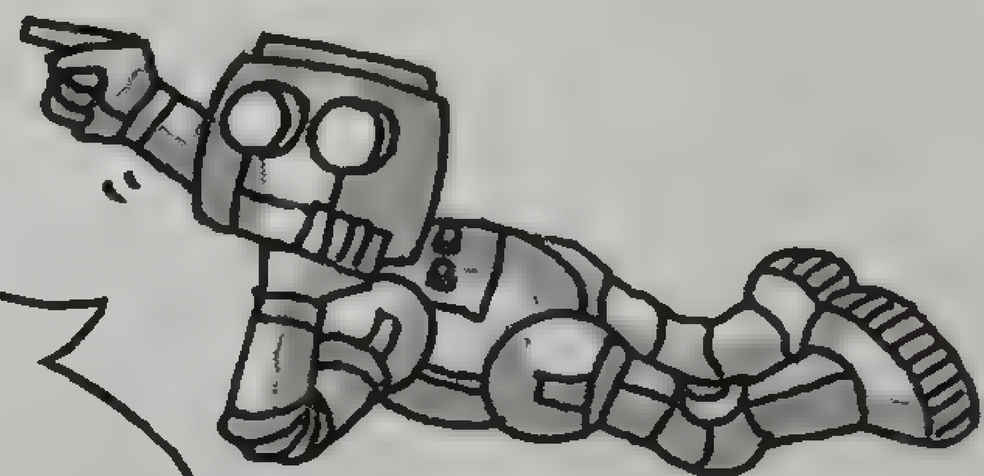
Dibuja una línea a S(I) * Y.

```
310 NEXT I
```

Vuelve al principio del bucle a dibujar el siguiente par de barras.

Si tu computadora puede imprimir texto y posiciones de pixels puedes añadir estas líneas para nombrar los gráficos. Por ejemplo, para hacer esto en el micro BBC necesitarías las líneas dadas abajo.

```
305 VDU 5: MOVE (I*X), INT(B(I)*Y)+50: PRINT «B»
308 VDU 5: MOVE (I*X)-70, INT (S(I)*Y)+50: PRINT «S»
```



Programa de gráfica lineal

Para una gráfica lineal necesitas imprimir el primer punto del gráfico y, entonces, dibujar una línea a cada punto a lo largo de la gráfica. Para hacer esto necesitas bucles separados para cada sistema de ordenación. Para transformar el programa de diagrama de barras para que haga una gráfica lineal reemplaza las líneas 270 en adelante por las siguientes líneas.

```
270 PLOT INT(X), INT(B(1)*Y)
```

Dibuja el primer punto de la ordenación de burbuja X pixels a lo largo y B(1) *Y pixels hacia arriba.

```
280 FOR N=2 TO 5
290 DRAW INT(N*X), INT(B(N)*Y)
300 NEXT N
```

Bucle para dibujar la gráfica para la ordenación de burbuja. Cada vez que el bucle se repite, N crece 1 y la computadora dibuja una línea al siguiente punto a lo largo de la gráfica.

```
310 PLOT INT(X), INT (S(1)*Y)
```

Dibuja el primer punto para la Shell sort X pixels a lo largo y S(1) * Y pixels hacia arriba.

```
320 FOR N=2 TO 5
330 DRAW INT (N*X), INT(S(N)*Y)
340 NEXT N
```

Bucle para dibujar la gráfica para la Shell sort.

Más manejo de cadenas

El programa que hay en las próximas páginas hace que la computadora aparente tener una conversación contigo. Por supuesto, la computadora sólo es tan inteligente como el programa que le introduzcas y todas las palabras o frases de sus respuestas están almacenadas en matrices de cadena en el programa*.

La principal labor del programa es hacer que la computadora escoja las palabras correctas para contestarte. Contiene algunas rutinas BASIC de manejo de cadenas que hacen que sus respuestas parezcan, a veces, casi «inteligentes». El éxito de este tipo de programa reside no sólo en la estructura del programa, sino también en las palabras o frases que introduces en él. Podrían probar a cambiar el vocabulario de la computadora para hacer que hablara de diferentes temas, o hacer sus respuestas más amistosas... o antipáticas.

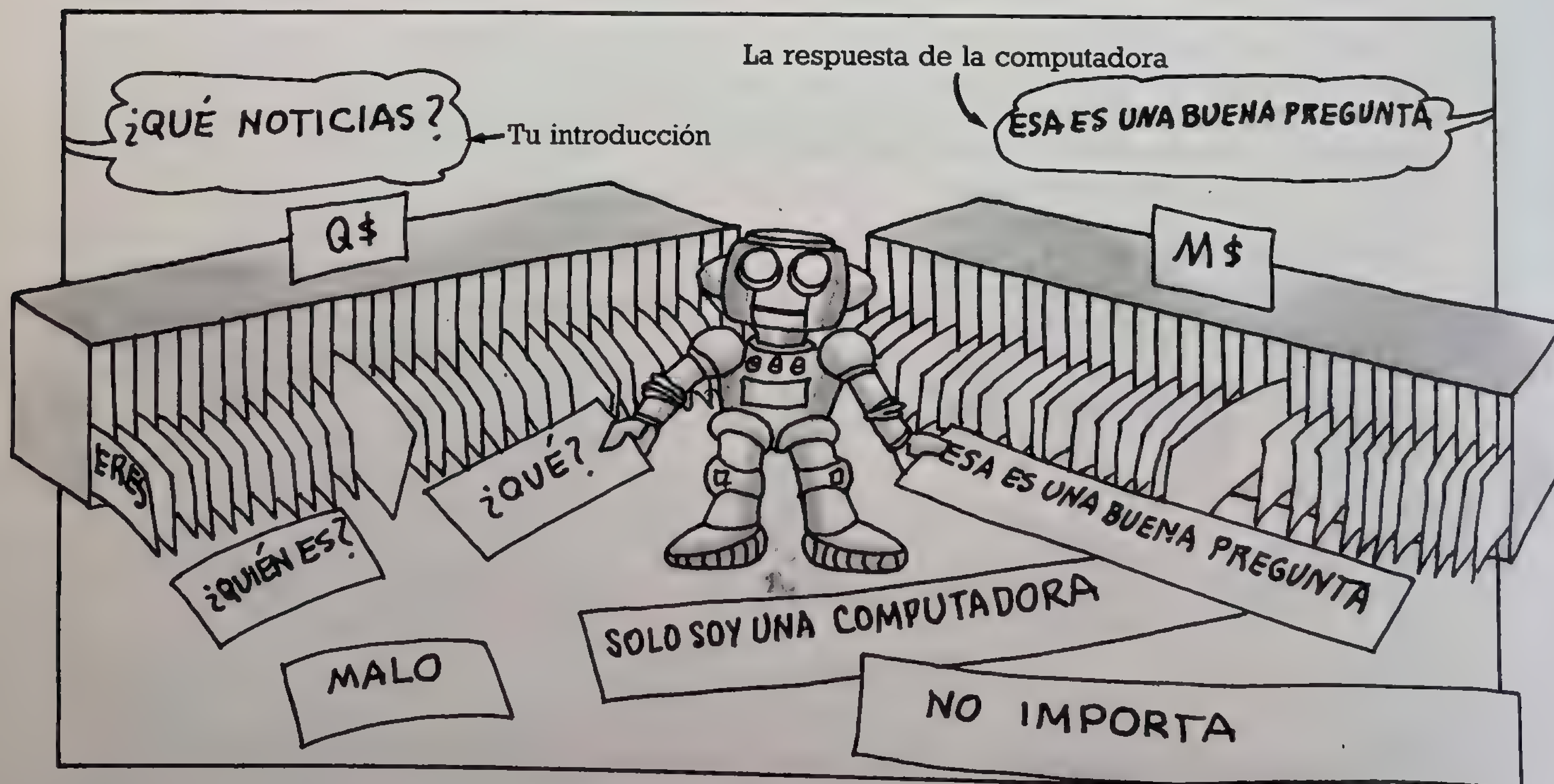
Ejecuciones ejemplo

```
HOLA, ¿COMO TE LLAMAS? JUDY
HABLAME, JUDY
HOLA COMPUTADORA
¿QUE PIENSAS SOBRE LAS NOTICIAS?
¿QUE NOTICIAS?
ESA ES UNA BUENA PREGUNTA
¿NO ME LO VAS A DECIR?
ESCUCHA JUDY, PIENSO QUE ERES TAN
AMISTOSA COMO LAS OTRAS PERSONAS
CON LAS QUE HE HABLADO
GRACIAS
NO LO MENCIONES
```

```
¿DE QUE QUIERES HABLAR?
HABLAME DEL TIEMPO, JUDY
ESTA LLOVIENDO
¿QUE HACE QUE ESTES SEGURA?
LO PUEDO VER
HE OIDO QUE TU ERES UNA ESPECIE
DE GENIO CHISTOSO, JUDY
¿QUIEN TE LO DIJO?
NO IMPORTA
¿POR QUE NO CONTESTAS A MIS PREGUNTAS?
MI QUERIDA AMIGA JUDY. ¿TU NO PIENSAS QUE
TODOS
LOS HUMANOS SON MALEDUCADOS? ¿NO?
```

Cómo funciona

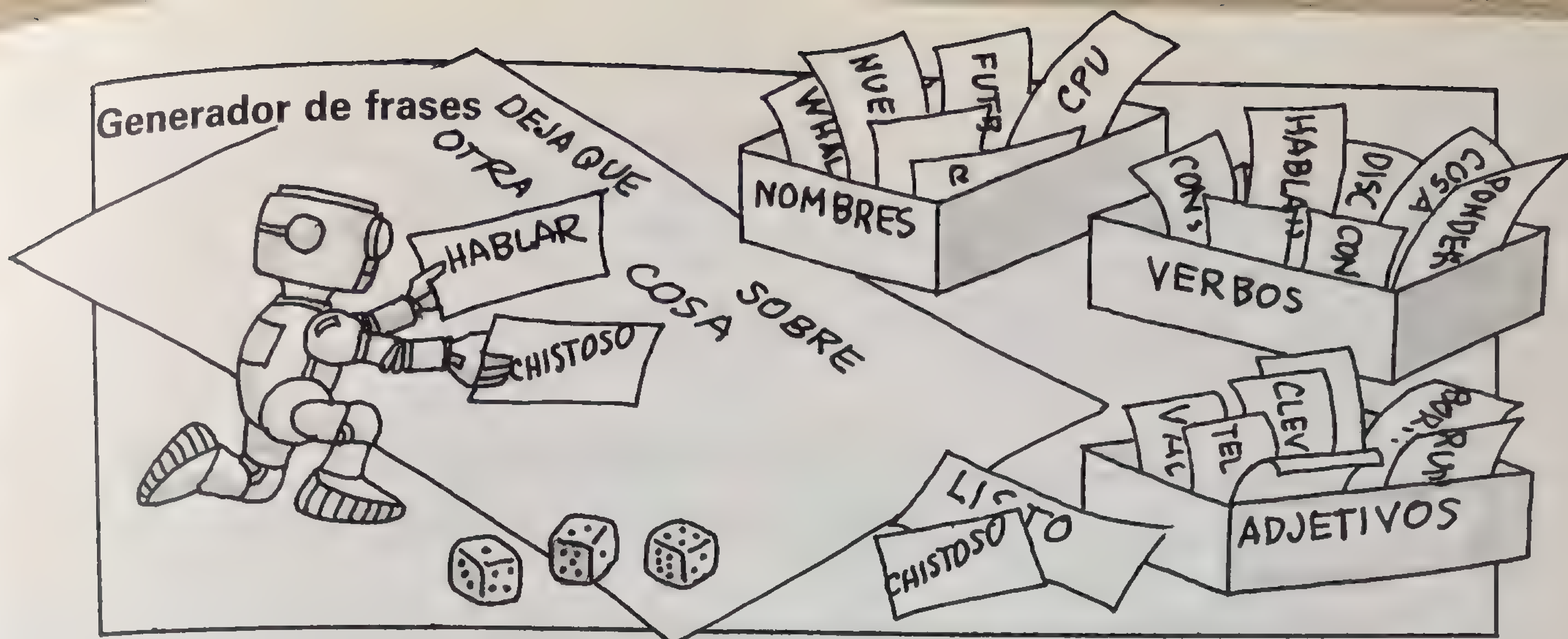
Hay dos métodos diferentes en el programa para producir las respuestas de la computadora. Una es una rutina de «comprobación de frases» y la otra es un generador de frases aleatorias.



La rutina de comprobación de frases tiene una lista de palabras usadas frecuentemente almacenada en una matriz llamada Q$. Para cada palabra o frase hay una respuesta adecuada almacenada en M$. Cuando tecleas algo, la computadora comprueba para ver si has usado alguna de las frases de Q$, y si lo has hecho, usa la correspondiente respuesta en M$.

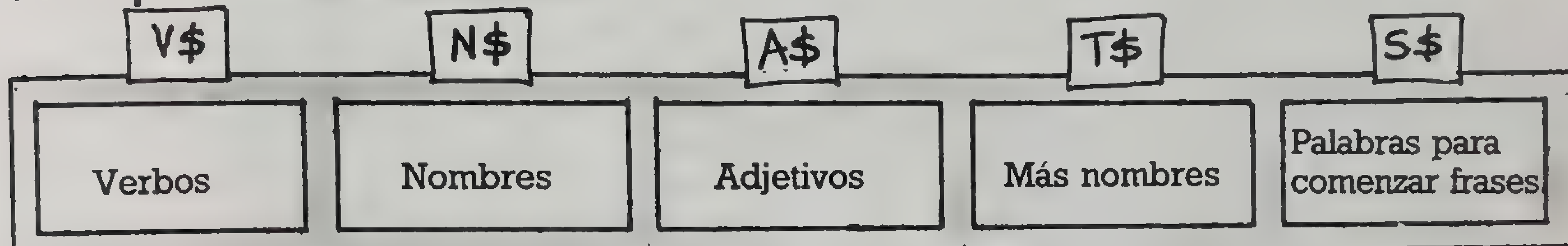
* Para transformar el programa para computadoras Sinclair (Timex), ver página 47.

Generador de frases

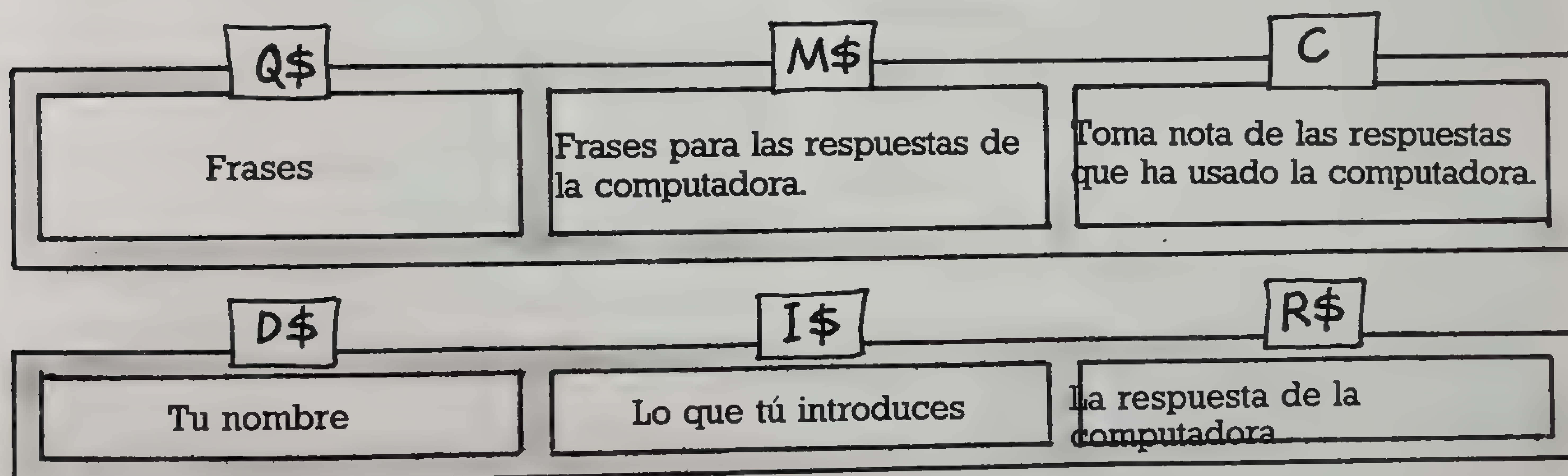


El generador de frases aleatorias consiste en frases a medio formar que las computadoras completas con verbos, nombres y adjetivos escogidos aleatoriamente. Todas las palabras se almacenan en matrices en la memoria de la computadora y han ido especialmente escogidas para que tengan sentido en las frases.

Para qué son las variables



Estas son las matrices para almacenar las palabras para las frases aleatorias.



El programa

```
100 CLS
110 DIM V$(10), N$(10), A$(10)
120 DIM T$(10), S$(10)
130 DIM M$(30), Q$(30), C(30)
140 REM MIRA EN LOS DATOS
150 GOSUB 1000
200 REM PERSON'S INPUT
210 INPUT «HOLA, ¿COMO TE LLAMAS?»; D$
220 PRINT
230 PRINT «HABLAME»; D$
240 INPUT I$
250 IF I$=« » THEN GOTO 220
260 IF I$=«ADIOS» THEN GOTO 910
```

Líneas 110–130: Fija las matrices para almacenar palabras y frases (N.B. En algunas computadoras no necesitas dimensionar matrices de menos de diez elementos).

Línea 150: Va a una subrutina para introducir todas las palabras y frases en las matrices.

Línea 240: Tu respuesta se almacena en I$.

Línea 250: Salvaguarda en caso de que el usuario presione RETURN directamente y I$ esté vacío.

Línea 260: Si tecleas ADIOS, la computadora va a la línea 910 a preguntarte si quieres dejar de ejecutar el programa.

EL LISTADO CONTINUA A LA VUELTA DE LA PAGINA

```
300 REM RESPUESTA DE LA COMPUTADORA
310 LET RESPUESTA=INT(RND(1)*8+1
320 IF RESPUESTA<6 THEN GOTO 490
```

El número aleatorio en la variable RESPUESTA decide qué método usará la computadora para responder. Si RESPUESTA es menor que 6 usa la rutina de comprobación de frase que empieza en la línea 490.

```
330 GOTO 600
```

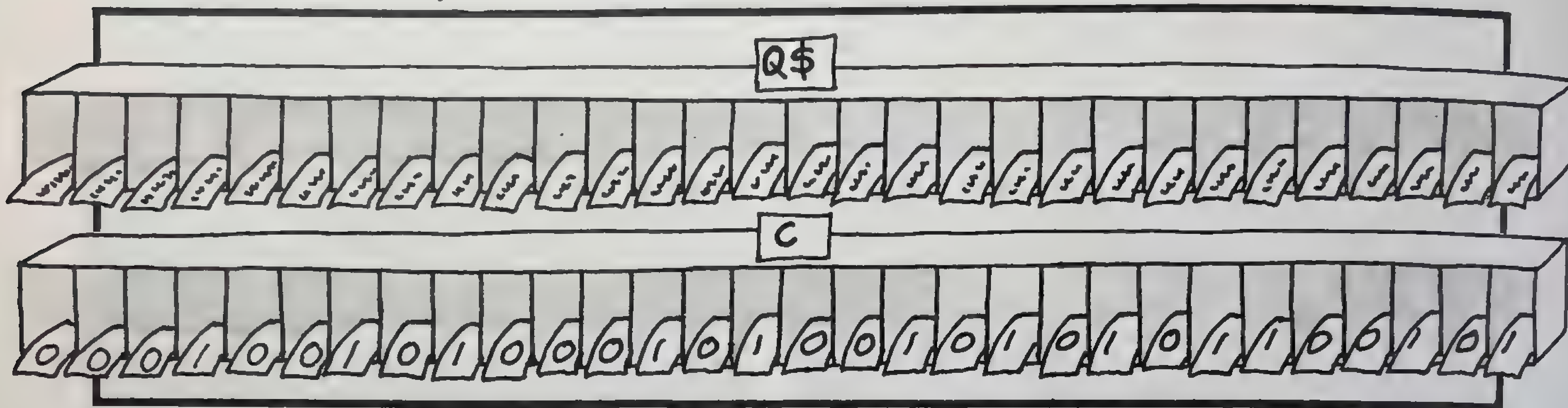
Si RESPUESTA es 6 o por encima va al generador de frases en la línea 600.

```
340 PRINT
350 PRINT R$
360 PRINT
```

Después de construir su respuesta la computadora la almacena en R$ y entonces la imprime en la pantalla en la línea 350.

```
400 REM COMPRUEBA CUANTAS RESPUESTAS SE HAN USADO
```

EL LISTADO CONTINUA ABAJO



Las líneas 400-470 comprueban cuántas respuestas de M$ han sido usadas. Cada vez que la computadora encuentra una de las frases de Q$ en lo que introduces, pone un número 1 como un marcador en la posición correspondiente en una matriz llamada C. Esto evita que use la respuesta para esa frase otra vez. Las líneas 400-470 comprueban cuántos marcadores hay en C y si hay más de 12, vuelven todos los marcadores a cero.

```
405 LET T=0
410 FOR K=1 TO 30
420 LET T=T+C(K)
430 NEXT K
```

Bucle para contar los marcadores que hay en la matriz C. El total es almacenado en T.

```
440 IF T<12 THEN GOTO 460
```

Si T es menor que 12, se han usado menos de 12 respuestas y la computadora no pone a cero los marcadores.

```
450 FOR K=1 TO 30: LET C(K)=0: NEXT K
```

Bucle para volver cada número a 0.

```
460 LET T=0
```

La variable T (la variable para contar los marcadores) es puesta a 0.

```
470 GOTO 240
```

Vuelta a la línea 240 para esperar a que la persona introduzca algo.

```
490 REM RUTINA DE COMPROBACION DE FRASES
500 FOR FRASE=1 TO 30
```

Bucle a ejecutar tantas veces como frases tenga Q$*.

```
510 LET L1=LEN(Q$(FRASE))
```

Cada vez que el bucle se repite, la computadora mide la longitud de la frase siguiente en Q$ y almacena la longitud en LI.

```
520 LET L2=LEN(I$)
```

El número de caracteres que tu introduces (I$) se almacena en L2.

```
530 FOR TEST=1 TO L2
540 IF MID$(I$, TEST, L1)=Q$
    (FRASE) THEN GOTO 560
```

TEST es un bucle de anidamiento a ejecutar tantas veces como caracteres hay en lo que hayas introducido. Cada vez que el bucle se repite la computadora comprueba los caracteres en I$ con aquellos en la frase de Q$. Si coinciden va a la línea 560.

```
550 NEXT TEST: NEXT
    FRASE: GOTO 600
```

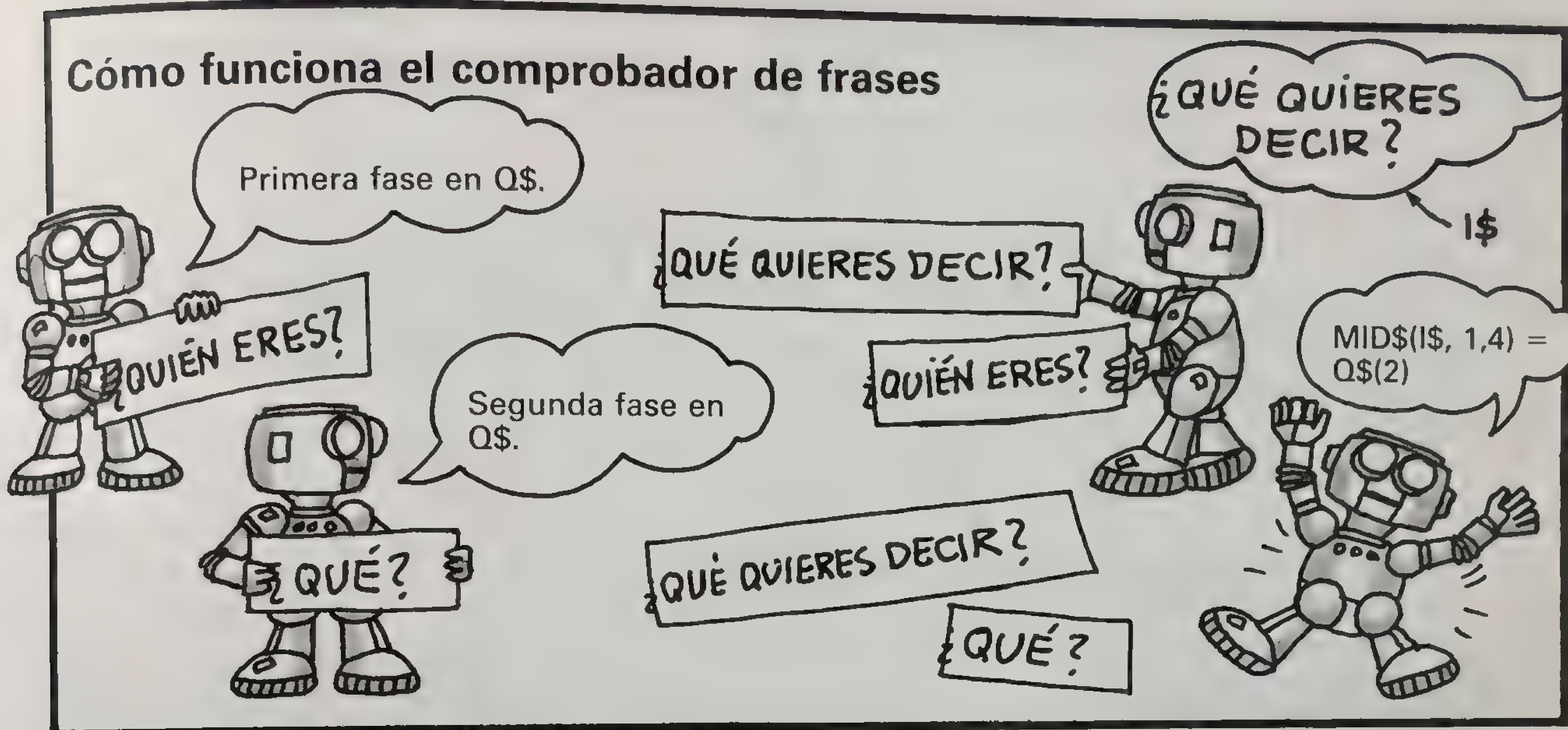
Si, después de repetir todos los bucles, la computadora no puede encontrar ninguna de las frases de Q$ en L$, va al generador de frases en la línea 600.

* Si estás usando un micro BBC, mira la nota en la página 46.

```
560 IF C(FRASE)>0 THEN GOTO 550
570 LET C(FRASE)=C(FRASE)+1
580 LET R$=M$(FRASE)
590 GOTO 340
```

Si encuentra una frase que coincide, salta fuera del bucle y viene a la línea 560. Entonces comprueba el marcador en la matriz C que corresponde a la frase. Si el marcador no es 0 vuelve a los bucles a ver si hay otra frase que coincide en I$. Si el marcador es 0 lo cambia a 1 en la línea 570. Entonces en la línea 580 busca la frase correspondiente en M$ y la pone en R$ lista para ser impresa en la línea 350.

EL LISTADO CONTINUA ABAJO



La primera vez que se efectúa la rutina de comprobación de frases FRASE = 1 por lo que la computadora examine la primera frase en Q$. La primera vez que se efectúa el bucle TEST, TEST = 1 así que la computadora compara la frase en Q$ con los caracteres 1 a 7 (la longitud de Q$) en I$. Si no son iguales repite el bucle TEST. Esta vez TEST = 2 así que compara los caracteres 2 a 8, y en adelante. Si los caracteres en I$ no son iguales a Q$ la computadora vuelve al principio del bucle FRASE a seleccionar la frase siguiente de Q$.

```
600 REM GENERADOR DE FRASES
610 LET E=INT(RND(1)*10+1)
620 LET F=INT(RND(1)*10+1)
630 LET G=INT(RND(1)*10+1)
640 LET H=INT(RND(1)*10+1)
650 LET L=INT(RND(1)*10+1)
660 ON E GOTO 700, 720, 740, 760, 780, 800, 830, 850, 870, 890
700 LET R$=«QUE PIENSAS
    SOBRE «+N$(H)+»?»
```

Números aleatorios para escoger qué palabras y frase utilizar. El número en E decide qué frase usará la computadora. F es para verbos, G es para adjetivos, H es para nombres y L es para comienzos de frase.

El número en E le dice a la computadora a qué línea ir. Si E = 1 va al primer número en la lista, si E = 2 va al segundo; etc. Para más sobre la palabra BASIC ON, ver página 20.

```
710 GOTO 340
720 LET R$=S$(L)+» «+D$+» «¿NO PIENSAS QUE TODOS LOS
    HUMANOS SON «+A$(G)+» NO?»
730 GOTO 340
740 LET R$=«HE OIDO QUE ERES UNA ESPECIE DE «+A$(G)+»
    «+T$(H)+» «+D$
750 GOTO 340
760 LET R$=S$(L)+» «+D$+», PIENSO QUE ERES TAN «+A$(G)»
```

D$ almacena tu nombre.

Las líneas 700-900 contiene diez frases parcialmente formadas que la computadora rellena con palabras de N$, V$, etc. Para hacer que la computadora sume cadenas de estas maneras usa un signo + . Tienes que tener cuidado de poner los espacios dentro de las comillas para que las frases estén debidamente espaciadas. La computadora pone la frase completa en R$ y entonces vuelve a la línea 340 para imprimirla.

EL LISTADO CONTINUA A LA VUELTA DE LA PAGINA

```
765 LET R$=R$+«COMO EL RESTO DE LA GENTE CON LA QUE HE HABLADO»
770 GOTO 340
780 LET R$=«ME SIENTO "+A$(G)+" AHORA»
790 GOTO 340
800 PRINT: PRINT «SSSHHH... ESTOY PENSANDO...»
810 LET R$=«DEJA QUE "+V$(F)+" "+N$(H)+" PIENSO "+N$(H)+" ES "+A$(G)
820 GOTO 340
830 LET R$=HABLAME DE "+N$(H)+", "+D$
840 GOTO 340
850 LET R$=«PIENSAS QUE SOY "+A$(G)+", "+D$+"?»
860 GOTO 340
870 LET R$=DEJA QUE "+V$(F)+" ALGUNA OTRA COSA "+A$(G)
880 GOTO 340
890 LET R$=«ADIVINA QUE ESTOY PENSANDO» +D$
900 GOTO 340
910 REM RUTINA DE DESPEDIDA
920 PRINT «¿YA HAS TENIDO BASTANTE?»
930 PRINT «¿HAY ALGUIEN AQUI QUE PUEDA HABLAR...?»
940 INPUT Z$: IF Z$=«SI» GOTO 210
950 PRINT: PRINT «ADIOS ENTONCES»
960 END
```



Si tu respuesta a la computadora es ADIOS, la línea 260 manda a la computadora a esta línea.

```
1000 REM FRASES PARA LA RUTINA DE COMPROBACION DE FRASES
1010 FOR I=1 TO 30: READ Q$(I): NEXT I
1020 DATA QUIEN ERES, ¿QUE?, SIGINIFICA, PORQUE, TU
1030 DATA «YO», «EL», «HABLAR», «NO»
1040 DATA «ERES», «MI», «SI», TU, ?
1050 DATA «PIENSA, MALEDUCADO, GRACIAS, FUERA»
1060 DATA ELLOS, ?, INFERIOR, «NO», «ES»
1070 DATA A, ?, AHORA
```

Estas líneas contienen las frases que la computadora busca en la que tú introduces. Ten cuidado en escribirlas exactamente como están aquí ya que los espacios de dentro de las comillas son parte de los datos.

```
1100 REM RESPUESTAS DE LA COMPUTADORA A LAS FRASES EN Q$
1110 FOR I=1 TO 30: READ M$(I): NEXT I
1120 DATA SOLO SOY UNA COMPUTADORA
1130 DATA NO IMPORTA, ESA ES UNA
     BUENA PREGUNTA
1140 DATA NO LO SE, «¿BIEN, POR QUE NO?»
1150 DATA QUE QUIERES DECIR, «¿QUIEN ERES?»
1160 DATA OH, ¿QUE QUIERES DECIR?
1170 DATA QUIERES QUE ME CALLE
1180 DATA DIMELO TU, ¿QUE QUIERES DECIR?
1190 DATA MY-MY-MY, ASI QUE ESTAS DE ACUERDO
1200 DATA ¿NO TE GUSTO?
1210 DATA ¿POR QUE?, ACLARATE LAS IDEAS. GRACIAS
1220 DATA AUN NO HAS VISTO NADA
1230 DATA NO LO MENCIONES
1240 DATA Y TU, NO ME IMPORTA, QUE
     PREGUNTA MAS ESTUPIDA
1250 DATA TIENES UN COEFICIENTE INTELECTUAL BAJO, BASURA
1260 DATA QUE HACE QUE ESTES SEGURO, VETE
1270 PIERDETE, EL CONOCIMIENTO ES UN PROBLEMA PARA MI
```

Estas son las respuestas de la computadora para cada una de las frases en Q$. Las respuestas están en el mismo orden que las frases. Por ejemplo, el quinto elemento en M$ es la respuesta a la quinta frase en Q$.

```
1300 REM BUSCA EN LOS NOMBRES
1310 FOR I=1 TO 10: READ N$(I): NEXT I
1320 DATA FUTBOL, BAILE
1340 DATA EL TIEMPO, LAS NOTICIAS
1350 DATA MI CPU PESCANDO
1360 DATA LA BALLENA AZUL, EVOLUCION
1370 DATA GEOGRAFIA, COMIDA
```

Estos son los nombres a poner en las frases aleatorias.

```
1400 REM BUSCA EN LOS VERBOS
1410 FOR I=1 TO 10: READ V$(I): NEXT I
1420 DATA PIENSA SOBRE, HABLA SOBRE
1430 DATA DISCUTE, CONTEMPLA
1440 DATA REFLEJAR EN, MEDITAR SOBRE
1450 DATA PENSAR, PONDERAR
1460 DATA CERCIORATE, CONSIDERAR
1500 REM BUSCA EN LOS ADJETIVOS
1510 FOR I=1 TO 10: READ A$(I): NEXT I
1520 DATA ESTUPIDO, LISTO
1530 DATA INTELIGENTE, SABIO
1540 DATA CHISTOSO, AMISTOSO
1550 DATA TEDIOSO, PESADO
1560 DATA MALEDUCADO, VERSATIL
1600 REM BUSCA EN LOS COMIENZOS DE FRASES Y
     EN OTROS NOMBRES, POR PAREJAS
1610 FOR I=1 TO 10: READ S$(I), T$(I): NEXT I
1620 DATA DIOS MIO, ABURRIMIENTO
1630 DATA «BIEN», SAPO
1640 DATA VEAMOS, JOVEN PROMETEDOR
1650 DATA ESCUCHA, GENIO
1660 DATA MIRA, DUMBO
1670 DATA UMMM..., IMBECIL
1680 DATA AHORA, PARASITO
1690 DATA REALMENTE, PRODIGIO
1700 OH NO!, MONSTRUO
1710 DATA MI QUERIDO AMIGO, CAPRICHO DE COMPUTADORA
1720 RETURN
```

Estos son los verbos para las sentencias aleatorias.



Estos son los adjetivos.



Cada par de elementos de información consiste en una palabra para empezar una frase y un nombre. Son leidos en parejas para ahorrar espacio.

Ideas para cambiar el programa

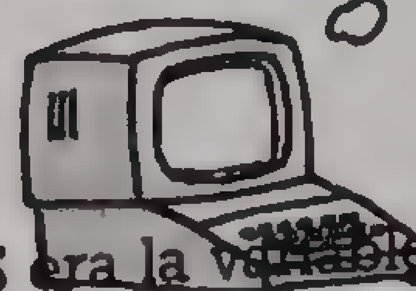
1. La manera más fácil de cambiar este programa es cambiar las palabras y frases. Es mejor comprobar cada palabra en cada frase para asegurarse que tienen sentido. Por el momento, todos los nombres en N$ son singulares. Si usas nombres en plural, necesitarás cambiar los verbos. Puedes añadir palabras si quieres. Si lo haces necesitarás cambiar el tamaño de las matrices. Los bucles para buscar en los datos y los números aleatorios en las líneas 610-650.

2. También podrías probar a cambiar las palabras en Q$ para hacer que la computadora saque frases diferentes. Necesitarás pensar respuestas adecuadas para cada nueva frase y poner las respuestas en las posiciones correctas en M$.

3. Para hacer que la computadora use el generador de frases aleatorias más frecuentemente cambia el número 6 en la línea 320 por un número más bajo. También puedes cambiar la frecuencia con que la computadora vuelve a 0. Los marcadores de respuesta en la matriz C. Para hacer esto, cambia el número 12 en la línea 440.

Modalidad de soñar despierto

Puedes hacer que la computadora hable consigo misma añadiendo las siguientes líneas al programa:




```
160 PRINT «MODALIDAD DE CONVERSACION
    O SOÑAR DESPIERTO
170 INPUT K$
180 IF K$=«D» THEN LET D$=
    «ROM»: GOTO 600
470 IF K$=«C» THEN GOTO 240
475 IF K$=«D» THEN LET I$=R$
480 LET R$=« »: GOTO 310
```

D$ era la variable para almacenar tu nombre. En la modalidad de soñar despierto la computadora usa la palabra ROM como nombre para sí misma, entonces va a la línea 600 a generar una frase aleatoria.

La respuesta de la computadora R$ se vuelve la nueva introducción I$. Entonces la computadora vuelve a la línea 310 a escoger su método de respuesta.

Rutina de respuesta

En estas dos páginas hay otra rutina que puedes añadir al programa de la conversación. Hace que la computadora te responda usando tus mismas palabras. Las conversaciones para computadoras Sinclair se dan al final de la página opuesta.



La rutina de respuesta funciona de una manera similar a como lo hace la rutina de comprobación de frases. Hay dos matrices de información U$ y W$. U$ contiene las frases que tu podrías utilizar en lo que introduces y W$ contiene las respuestas de la computadora. Si usas una de las frases de U$ la rutina de respuesta la reemplaza con la correspondiente frase de W$ y entonces le añade al resto de tu frase.

135 DIM U$(9), W$(9) 138 DIM Z$(5)	Fija las matrices para U$ y W$ y otra matriz llamada Z$ para almacenar las respuestas de la computadora.
155 GOSUB 2200	Va a la subrutina para buscar en los datos.
325 IF REPLY=7 THEN GOTO 2000	Le dice a la computadora cuándo usar la rutina de respuesta.
2000 REM RUTINA DE RESPUESTA	
2010 LET Z=0	Z es un contador.
2020 LET P=LEN(I$)	Esto es, el número de caracteres que introduces.
2030 FOR A=1 TO P	Bucle a efectuarse tantas veces como caracteres hay en lo que tu introduces.*
2040 FOR B=1 TO 9	Bucle a efectuarse tantas veces como caracteres hay en el elemento más largo en W$.
2050 LET L=LEN(U$(B))	Cada vez que se repite el bucle B, la longitud del siguiente elemento en U$ se pone en L.
2060 IF MID$(I$, A, L)=U$(B)THEN GOTO 2140	Compara los caracteres A a L de lo que tu introduces con la frase con el subíndice del valor de B en U$ (B es fijado por el bucle). Si las frases coinciden, la computadora va a la línea 2140.
2070 NEXT B: NEXT A	Cada vez que se repite el bucle B la computadora comprueba la frase siguiente en U$. Cuando se repite el bucle A recoge la siguiente secuencia de caracteres en I$.
2080 IF Z$(1)=« » THEN GOTO 600	Vuelta al generador de frases aleatorias si ninguna frase coincide.

* Si estás usando un micro BBC, ver la nota en la página 46.

```
2090 FOR J=1 TO Z
2100 PRINT Z$(J);
2110 LET Z$(J)=« »
2120 NEXT J
```

Imprime todas las respuestas en Z$.

```
2130 LET R$=I$: GOTO 350
```

Pone la parte que quede de tu frase en R$ y entonces vuelve a la línea 350 a imprimirla.

```
2140 LET Z=Z+1
```

Z cuenta las respuestas en Z$.

```
2150 IF A>1 THEN LET Z$(Z)=LEFT$
     (I$, A−1)+« »+W$(B)+« »
```

La variable A fija el bucle en la línea 2030 y es el número del primer carácter de la frase en I$ que coincide con la frase en U$. Si A > 1 la línea 2150 pone los caracteres a la izquierda de la frase en Z$ y entonces le añade la respuesta de W$.

```
2160 IF A<2 THEN LET Z$(Z)=W$(B)+« »
```

Si A < 2 entonces la frase que coincide está al principio de I$ así que la computadora sólo pone su respuesta en Z$.

```
2170 LET I$=MID$(I$, A+L, P)
```

Pone el resto de tu frase en Z$ y entonces vuelve a los bucles a ver si hay otra frase que coincida.

```
2180 GOTO 2020
2200 REM DATOS PARA LA RUTINA DE RESPUESTA
2210 FOR I=1 TO 9
2220 READ U$(I), W$(I)
2230 NEXT I
2240 DATA YO SOY, TU ERES, TU ERES, YO SOY
2250 DATA «YO», TU, «MI», TU
2260 DATA «YO», «TU», «TUYO», «MIO»
2270 DATA «TUYO», «MI», «MIO», «TUYO»
2280 DATA TU, COMPUTADORA
2290 RETURN
```

Estos son los datos para U$ y W$.

Rutina de respuesta para computadoras Sinclair

Introduce las siguientes líneas para el Spectrum y el ZX81. De todas maneras para el ZX81 necesitarás usar el método dado en la página 47 en el programa de conversación para introducir los datos. Pon el bucle del ZX81 para introducir datos entre las líneas 1000 y 1720 y las sentencias DIM antes de la línea 1000.

```
 135 DIM U$(9, 7)
 136 DIM W$(9, 9)
 137 DIM Z$(5, 20)
2042 LET P$=« »
2044 FOR I=1 TO LEN (U$(B))
2046 IF U$(B) (I TO I)< >«*» THEN LET P$=P$+U$(B) (I TO I)
```

Pon una frase U$ en P$ usando las *s para encontrar el final de la frase.

```
2048 NEXT I
2050 LET L=LEN(P$)
2055 IF L+A−1>P THEN GOTO 2070
```

Si P$ es más larga que la frase I$, se vuelve atrás a seleccionar la frase siguiente.

```
2060 IF I$(A TO A+L−1)=P$ THEN GOTO 2140
```

Si la frase I$ = a la frase P$ va a la línea 2140.

```
2080 IF Z=0 THEN GOTO 600
```

Si ninguna frase coincide entonces vuelve a generador de frases aleatorias.

```
2150 IF A>1 THEN LET Z$(Z)=I$(TO A−1)+« »+W$(B)+« »
2160 IF A <2 THEN LET Z$(Z)=W$(B)+« »
```

Si A > 1 pone los caracteres al principio de I$ en Z$ y añade la frase W$. Si A < 2 sólo pone la frase W$ en Z$.

```
2170 LET I$=I$(A+L TO)
2240 DATA «YO SOY***», «TU ERES», «TU ERES», «YO SOY»
2250 DATA «YO *****», «TU», «MI****», «TU»
2260 DATA «YO ***», «TU», «TUYO *», «MIO»
2270 DATA «TUYO *», «MI», «MIO**», «TUYO»
2280 DATA «YO ***», «COMPUTADORAS»
2290 RETURN
```

Los espacios son una importante parte de la importante información, así que tienes que completar las cadenas con *s.

Transformación de programas

Estas dos páginas te enseñan cómo transformar los programas para que funciones en ZX81 y Spectrum y cómo transformar los programas de gráficos para que funcionen en computadoras que dibujen líneas en relación con el último punto dibujado. Al igual que insertar las líneas que aquí se dan, también tienes que hacer los otros cambios necesarios a tu computadora, por ejemplo, usar el comando de gráficos de tu computadora. La instrucción RND y cambiar los nombres de las variables si es necesario.

Juego de encontrar palabras para computadoras Sinclair (Timex).

Cambia BUSCANDO$ por C$ y cambia las líneas 170 y 200 por las siguientes.

```
170 LET L$=W$(R TO R)
200 LET C$=C$(2 TO)+L$
```

Base de datos para Spectrum (Timex 2000)

```
10 DIM T$(10, 14): DIM Y(12):
15 DIM M(10, 12)
120 ┐
417 ├── Igual que ZX81
425 ┘
```

Recuerda poner entre comillas cada elemento de información en las líneas DATA.

Base de datos para ZX81 (Timex 1000)

```
10 DIM T$(10, 14)
12 DIM Y(12)
14 DIM M$(10, 12)
120 GOSUB 100+100*C
417 LET L=LEN(Z$)
425 IF Z$=T$(I) (1 TO L) THEN GOTO 440
455 IF VAL(M$(I, J))=1 THEN PRINT «GANO
    LA COPA DEL MUNDO EN»; Y(J)
460 IF VAL(M$(I, J))=2 THEN PRINT «FUE
    FINALISTA EN»; Y(J)
555 IF VAL(M$(J, I)=1 THEN PRINT T$(J); «GANO
    LA COPA»
2170 IF C>0 AND C<6 THEN GOTO 2180
2175 PRINT «POR FAVOR,INTRODUCE UN NUMERO ENTRE 1 Y 5»
2176 GOTO 2150
```

Líneas 10, 12 y 14: El segundo subíndice es la longitud de la cadena más larga.

Línea 120: Usa C para calcular el número de línea.

Líneas 417 y 425: Le dice a la computadora qué caracteres comprobar.

Líneas 455 a 555: La matriz de datos se almacena en una matriz de cadena así que necesitas VAL para decirle a la computadora que tome el valor numérico.

En vez de tener que introducir todos los años puedes hacer que la computadora los calcule con las siguientes líneas.

```
1000 FOR I=1 TO 3
1010 LET Y(I)=1926+I*4
1020 NEXT I
1030 FOR I=0 TO 8
1040 LET Y(I+4)=1950+4*I
1050 NEXT I
```

Cambia las líneas 1100 a 1120 por diez sentencias LET, e.g.

```
1100 LET T$(1)=«URUGUAY»
1110 LET T$(2)=«ARGENTINA»
```

Cambia las líneas 1200 a 1310 por diez sentencias LET más e.g.

```
1200 LET M$(1)=«100100000000»
1210 LET M$(2)=«200000000010»
```

Añade una nueva línea:

```
1310 RETURN
```

Base de datos y programa de conversación para BBC.

Estos dos programas usan bucles para buscar en los datos, entonces saltan de los bucles cuando el elemento de información ha sido encontrado. El micro BBC sólo te dejará hacer esto diez veces al final de los cuales obtendrás el mensaje de error «TOO MANY FORS» (Demasiados FORS). Para evitar estos, cambia los bucles por variables que sean contadores con sentencias IF...THEN. Por ejemplo, usa las siguientes líneas en la base de datos:

```
420 LET I=0
422 LET I=I+1
430 IF I<10 GOTO 422
520 LET I=0
522 LET I=I+1
530 IF I<12 GOTO 522
```

Necesitarás hacer lo mismo para las líneas 500 a 550 del programa de conversación y las líneas 2030 a 2070 de la rutina de respuesta.

Gráficos instantáneos y dibujo de gráficos

Para computadoras (por ejemplo, Spectrum y Oric) que dibujan líneas a un punto X, Y medido desde el punto previamente dibujado y no desde la esquina de la pantalla cambia las siguientes líneas en los programas de Gráficos instantáneos y de gráficos (Necesitarás cambiar DRAW y PLOT con los comandos de gráficos de tu computadora.

Gráficos instantáneos

```
175 DRAW CX(1)−CX(4), CY(1)−CY(4)
180 FOR I=2 TO 4
185 DRAW CX(I)−CX(I−1), CY(I)−CY(I−1)
530 DRAW LX(ROW+2, I)−LX(ROW, I),
    LY(ROW+2, I)−LY(ROW, I)
```

Para encontrar las coordenadas del final de la línea, la computadora resta las coordenadas del último punto dibujado.

Diagrama de barras

```
220 PLOT 1, H: DRAW 0, −H+1: DRAW W, 0
300 DRAW 0, INT (B(I)*Y)
320 DRAW 0, INT(S(I)*Y)
```

Gráfica lineal

```
290 DRAW X, INT((B(N)−B(N−1))*Y)
330 DRAW X, INT((S(N)−S(N−1))*Y)
```

Programa de conversación para Spectrum (Timex 2000)

Haz los siguientes cambios para el Spectrum:

```
110 DIM V$(10, 11): DIM N$(10, 16): DIM A$(10, 11)
120 DIM T$(10, 14): DIM S$(10, 14)
130 DIM M$(30, 29): DIM Q$(30, 7) DIM C(30)
245 PRINT I$
500 FOR Q=1 TO 30                        ——— Usa Q en vez de FRASE.
510 LET P$=« »
512 FOR I=1 TO LEN (Q$(Q))
514 IF Q$(Q) (I TO I)< >« » THEN
    LET P$=P$+Q$(Q) (I TO I)             ——— Pone la frase de Q$ en P$.
516 NEXT I
530 FOR T=1 TO L2−LEN(P$)+1
540 IF I$(T TO T+LEN(P$)−1)=P$ THEN
    GOTO 560                             ——— Comprueba si I$ = P$.
550 NEXT T: NEXT Q: GOTO 600
560
570    Cambiar la variable FRASE a Q
580
660 GOTO ((E<7)*(680+E*20))+((E>6)*
    (810+(E−6)*20))                      ——— Esto significa que si E < 7
                                             GOTO (ve) a la línea número
                                             680 + E*20 y si E > 6 GOTO
                                             (ve) a la línea número
                                             (810 +− 6)*20).
```

Programa de conversación para ZX81 (Timex 1000)

Para el ZX81 necesitas un sistema para introducir todos los datos. Puedes hacer esto con sentencias INPUT y bucles. Para ejecutar el programa en ZX81 haz los siguientes cambios:

1. Haz los mismos cambios dados para el Spectrum arriba, pero pon las sentencias DIM en las líneas 970-990.

2. Cambia las líneas READ/DATA por sentencias INPUT, ej.:

```
1000 REM FRASES PARA LA RUTINA DE COMPRO-
     BACION DE FRASES
1010 FOR I=1 TO 30
1020 INPUT Q$(I)
1030 NEXT I
```

3. Cambia la línea 1720 para que sea:

```
1720 STOP
```

4. Teclea el programa, entonces teclea RUN 970 e introduce toda la información a medida que la computadora te la pide.

5. A continuación, para probar el programa, teclea GOTO 100. No presiones RUN, ya que si los hace toda la información se perderá.

6. Ahora puedes grabar el programa en un cassette. Cuando lo recojas, teclea siempre GOTO 100 para ejecutar el programa.

Respuestas

Problema del robot corredor (pág. 24)

```
10 INPUT «¿CUAL ES LA TEMPERATURA?»; TEMP
20 INPUT «¿CUANTOS SEGUNDOS?»; S
30 IF TEMP< >60 THEN LET D=S*(500+10*(TEMP-60))
   ELSE LET D=500*S
40 IF D<1 THEN PRINT DEMASIADO FRIO PARA ZAK
50 PRINT «A»; TEMP; «GRADOS, ZAK
   PUEDE CORRER»; D; «METROS EN»; S; «SEGUNDOS»
```

La computadora calcula primero 10*(TEMP −y esto le da la diferencia de distancia para TEMP grados (Si TEMP está por debajo de 60, la respuesta de este cálculo es negativa). Sumando 500 a la respuesta da la distancia que Zak puede correr en un segundo y multiplicándola por S da la distancia en S segundos.

Cambios en el Shell sort (pág. 34)

```
90 LET X=0
95 LET CAMBIO=0
231 LET X=X+1
271 LET CAMBIO=CAMBIO+1
365 PRINT «HUBO»; X; «COMPARACIONES
    Y»; SWAP; «CAMBIOS»
```

Barras más anchas (pág. 37)

```
285 FOR J=1 TO 8 STEP 2
290 PLOT INT (I*X+J), 1
300 DRAW INT(I*X+J), INT (B(I)*Y)
310 PLOT INT(I*X-4-J), 1
320 DRAW INT(I*X-4-J), INT(S(I)*Y)
325 NEXT J
```

Puede que necesites cambiar los números para que le sirvan a tu computadora.

Indice

Impreso en España - Printed in Spain. MELSA. Pinto (Madrid).
Depósito legal: M-43512-1985
I.S.B.N. 84-7374-145-5

Colección Electrónica

Colección Electrónica
PROGRAMACION DE COMPUTADORAS
Programas en BASIC



Colección Electrónica
MICRO COMPUTADORAS



Colección Electrónica
CALCULADORAS DE BOLSILLO



Colección Electrónica
JUEGOS DE COMPUTADORAS



Colección Electrónica
MANUAL DE GRABACION
CASSETTES



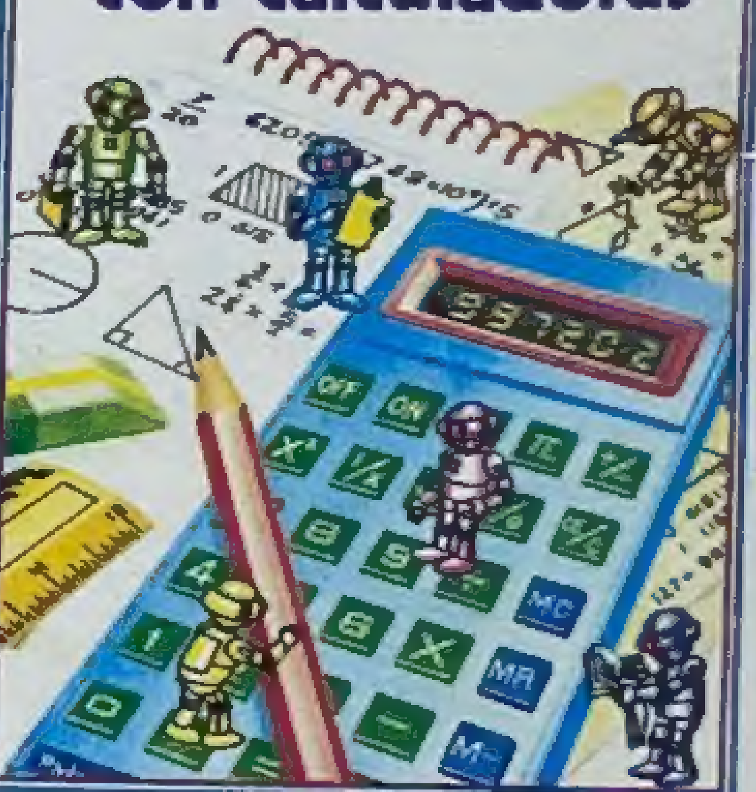
Colección Electrónica
PRACTICAS con COMPUTADORAS



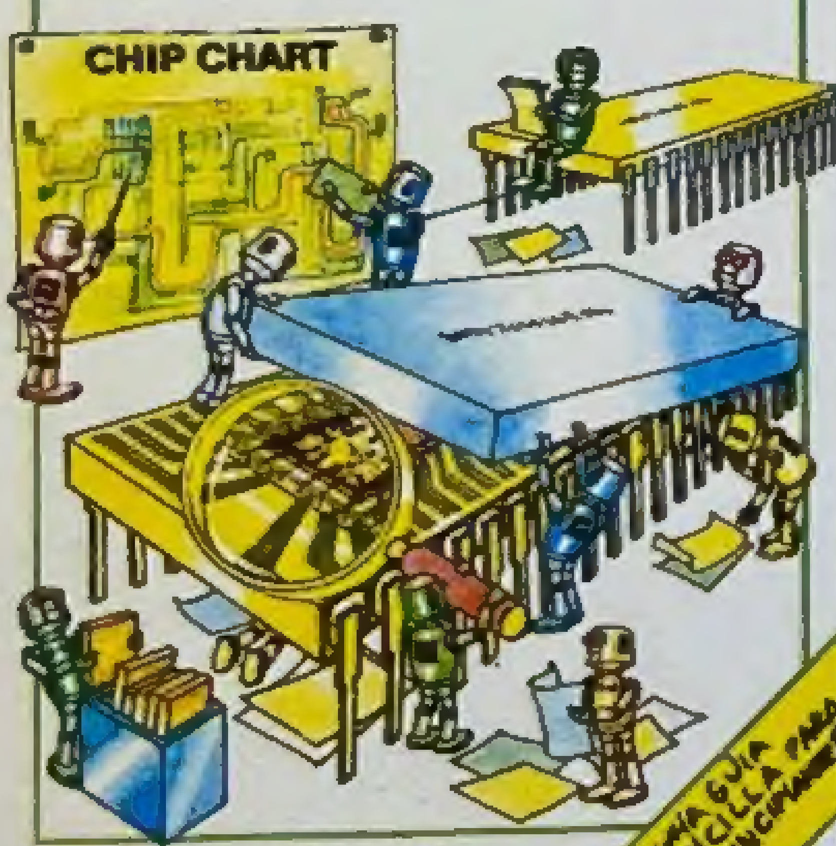
Colección Electrónica
PROGRAMA TUS PROPIAS **AVENTURAS** EN TU COMPUTADORA



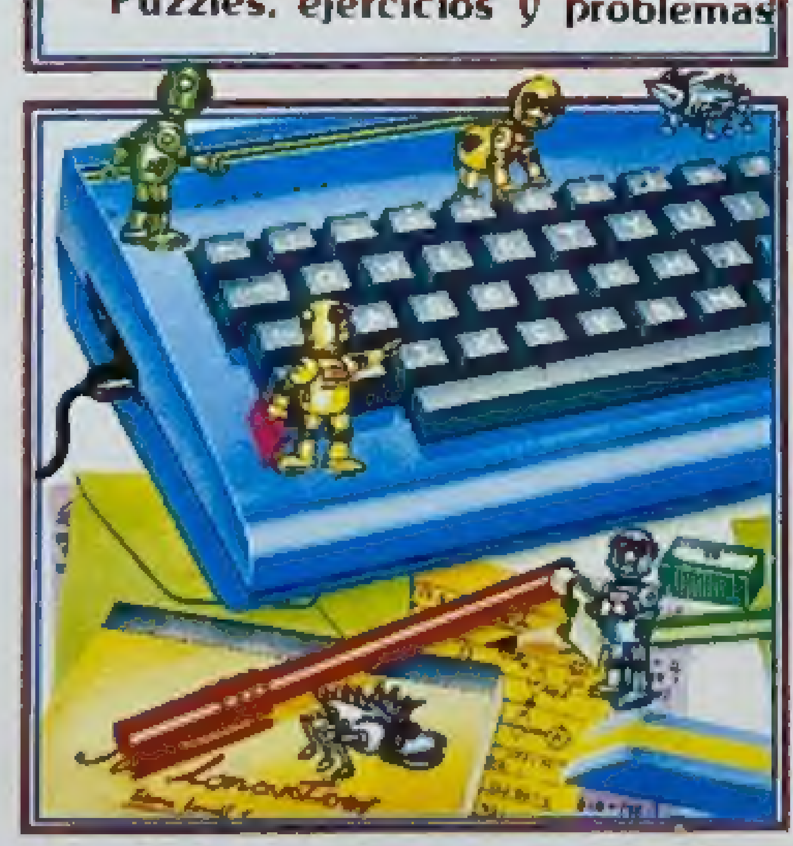
Colección Electrónica
CALCULOS Y HABILIDADES
con calculadoras



Colección Electrónica
EL CHIP
Qué es y qué puede hacer



Colección Electrónica
BASIC
Puzzles, ejercicios y problemas



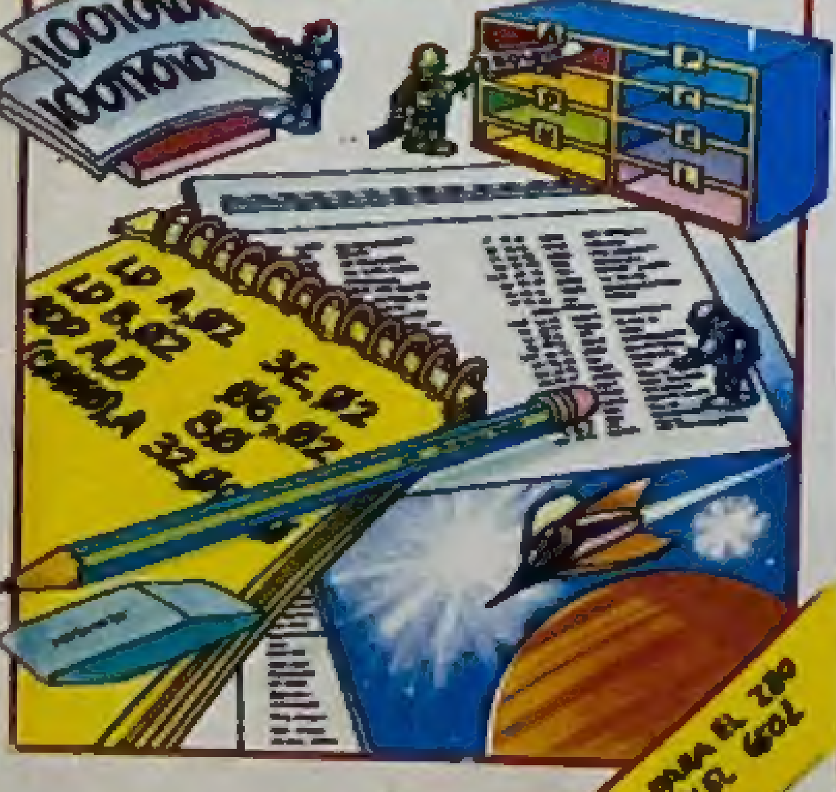
Colección Electrónica
EL LENGUAJE de las COMPUTADORAS



Colección Electrónica
DIBUJOS Y ANIMACION
CON COMPUTADORA



Colección Electrónica
CODIGO DE MAQUINA
PARA PRINCIPIANTES



distribuidor exclusivo
cesma sa
C/ Aguacate, 25
28044 MADRID

0719--788
065 421
El Corte Inglés

Colección Electrónica
GUIA DEL BASIC

